Available online at www.sciencedirect.com

SCIENCE DIRECT®

Discrete Optimization 2 (2005) 362–390

DISCRETE OPTIMIZATION

www.elsevier.com/locate/disopt

Geometric quadrisection in linear time, with application to VLSI placement

Jens Vygen*

Research Institute for Discrete Mathematics, University of Bonn, Lennéstraße 2, 53113 Bonn, Germany

Received 12 September 2003; received in revised form 15 August 2005; accepted 19 August 2005
Available online 17 October 2005

Abstract

We consider the following problem: given a set of points in the plane, each with a weight, and capacities of the four quadrants, assign each point to one of the quadrants such that the total weight of points assigned to a quadrant does not exceed its capacity, and the total distance is minimized.

This problem is most important in placement of VLSI circuits and is likely to have other applications. It is NP-hard, but the fractional relaxation always has an optimal solution which is "almost" integral. Hence for large instances, it suffices to solve the fractional relaxation. The main result of this paper is a linear-time algorithm for this relaxation. It is based on a structure theorem describing optimal solutions by so-called "American maps" and makes sophisticated use of binary search techniques and weighted median computations.

This algorithm is a main subroutine of a VLSI placement tool that is used for the design of many of the most complex chips.

Keywords: VLSI placement; Generalized Assignment Problem; Multiple Knapsack Problem; Quadrisection; Weighted median

1. Introduction

In many applications a set of objects has to be partitioned into a fixed number of subsets under capacity constraints. One example, which motivated the research which led to this paper, is the placement of very large-scale integrated (VLSI) circuits. Here, we have a Quadratic Assignment Problem, which is extremely hard to solve. It is standard practice to start by partitioning the set of components to be placed into four sets and assign each set to one quarter of the chip area.

As general objective functions are hard to deal with one often considers modular cost functions: for each component c we have a cost $r(c, i) \in \mathbb{R}$ if c is assigned to the ith set of the partition. The resulting Multiple Assignment Problem is a special case of the Generalized Assignment Problem [18] and a generalization of the Multiple Knapsack Problem [7]. It is naturally related to scheduling unrelated parallel machines [10,16].

For example, in VLSI placement the following situation occurs quite naturally [21,22]: each component has a position in the plane, and we want to move them as little as possible in order to meet the capacity constraints of the four quarters of the chip.

* Corresponding author. Tel.: +49 228 738770; fax: +49 228 738771.
E-mail address: vygen@or.uni-bonn.de.

1572-5286/$ - see front matter © 2005 Elsevier B.V. All rights reserved.
doi:10.1016/j.disopt.2005.08.007

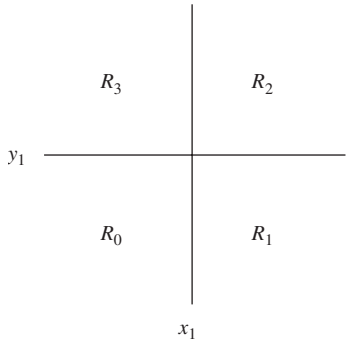

Fig. 1.

If we measure movement by weighted sum of ℓ_1-distances, or squared Euclidean distances, the cost function satisfies $r(c, 0) + r(c, 2) = r(c, 1) + r(c, 3)$ for all $c \in C$ (when numbering the regions counterclockwise; cf. Fig. 1). In this case, we speak of the Quadrisection Problem.

Even this special case of the Multiple Assignment Problem is NP-hard. However, if the size of the components is small compared to the capacities, we shall see that the fractional relaxation is very useful. We show that this leads to an efficient approximation algorithm for the Multiple Assignment Problem unless the capacity constraints are extremely tight. As a subroutine we solve the fractional relaxation by reduction to a Minimum Cost Flow Problem.

However, in the case of the Quadrisection Problem we can do better. In this important case, we describe an $O(|C|)$-algorithm for solving the fractional relaxation. In particular, this gives a best possible algorithm for the following problem: given a set C of points in the plane, each with a weight, and capacities of the four quadrants, assign each point to one of the quadrants such that the total weight of points assigned to a quadrant does not exceed its capacity, and the total movement is minimized; at most three points may be "split up" and partially assigned to several quadrants.

This paper is organized as follows: In Section 2, we consider the Multiple Assignment Problem and its fractional relaxation. We observe that there always exists an optimum fractional partition which is "almost" integral, and we describe an approximation algorithm except for very tight capacity constraints. In Section 3, we consider the special case arising in VLSI placement, the Quadrisection Problem. The rest of this paper then deals with the fractional relaxation of the Quadrisection Problem. A structure theorem, stating that there always exists an optimum fractional partition which is "consistent with an American map", is proved in Sections 5–7. Based on this proof, a linear-time algorithm (the main result of this paper) is described in Section 8. It uses a sophisticated binary search framework and weighted median computations as the main subroutines.

2. The Multiple Assignment Problem

The Generalized Assignment Problem asks for partitioning a finite set C (of components) to a fixed number of subsets under capacity constraints, minimizing a modular objective function. More precisely, we are given a finite set C and a positive integer m, weights $\text{size}(c, i) \geqslant 0$ for $c \in C$ and capacities $\kappa_i \geqslant 0$ for $i = 0, \ldots, m$. Moreover, we have a cost function $r : C \times \{0, 1, \ldots, m\} \to \mathbb{R}$. We look for a partition $f : C \to \{0, 1, \ldots, m\}$ meeting the capacity constraints $\sum\{\text{size}(c, j) : f(c) = j\} \leqslant \kappa_j$ $(j = 0, 1, \ldots, m)$ and minimizing $\sum_{c \in C} r(c, f(c))$.

This problem occurs in many contexts. As it contains the Knapsack Problem ($m = 1$), it is NP-hard. Even for $m = 1$ it is NP-complete to decide if a feasible solution exists (this contains the well-known decision problem Partition [11]), for variable m it is strongly NP-complete (it contains Bin Packing). On the other hand, the Generalized Assignment Problem can be solved in pseudopolynomial time for fixed m.

In many applications the resource consumption of each component is independent of the assignment, i.e. $\text{size}(c, 0) = \cdots = \text{size}(c, m) =: \text{size}(c)$. This is the special case that we call the Multiple Assignment Problem. It is also APX-hard [7]. However, if $\text{size}(c) \ll \min_{j=0}^{m} \kappa_j$ (as is often the case), the problem can be approximated well. Here, it is particularly interesting to consider the following fractional relaxation: find a fractional partition $g : C \times \{0, 1, \ldots, m\} \to [0, 1]$

with $\sum_{i=0}^{m} g(c,i) = 1$ $(c \in C)$ meeting the capacity constraints $\sum_{c\in C} \text{size}(c)g(c,i) \leqslant \kappa_i$ $(i = 0, 1, \ldots, m)$ and minimizing $\sum_{c\in C}\sum_{i=0}^{m} r(c,i)\ g(c,i)$.

The problem of finding an optimum fractional partition can be formulated as an LP, and in fact can be solved combinatorially by defining an equivalent minimum cost flow instance: define a digraph G by $V(G) := C \dot\cup \{s, t, 0, 1, \ldots, m\}$ and

$$E(G) := \{(s,c), (c,i), (i,t) : c \in C,\ i \in \{0, 1, \ldots, m\}\},$$

and capacities $u((s,c)) := \text{size}(c)$, $u((i,t)) := \kappa_i$ and $u((c,i)) := \infty$ for $c \in C$ and $i \in \{0, 1, \ldots, m\}$. The cost of an edge (c,i) is $r(c,i)$ $(c \in C,\ i \in \{0, 1, \ldots, m\})$, all other edges have zero cost. It is evident that maximum s–t-flows of minimum cost correspond to optimum fractional partitions.

Using the fastest known strongly polynomial algorithms for the general Minimum Cost Flow Problem [17,23] one can thus solve the problem (for constant m) in $O(n^2 \log^2 n)$ time. This has been improved by Tokuyama and Nakano [19], who obtained a running time of $O(n \log^2 n)$. If the values of $r(c,i)$ are integers with absolute value at most R, there exists an $O(n + \log R)$-algorithm [1], again for constant m.

The main reason for the interest in fractional partitions is that they often lead to good integral solutions. We mention three results in this spirit.

Proposition 2.1. *For any instance of the* Multiple Assignment Problem *there exists an optimum fractional partition g of C such that $g(c,i) \in \{0,1\}$ for all $c \in C\backslash C'$ and $i \in \{0, 1, \ldots, m\}$, with $|C'| \leqslant m$.*

Proof. Let g be an optimum fractional partition. Among those choose g such that $\Phi(g) := |\{(c,i) : c \in C,\ i \in \{0, 1, \ldots, m\},\ g(c,i) > 0\}|$ is minimal.

Let G be the undirected graph with vertex set $\{0, 1, \ldots, m\}$ that contains an edge $\{i, j\}$ for each $c \in C$ and each pair i, j with $i < j$, $g(c,i) > 0$, $g(c,j) > 0$ and $g(c,k) = 0$ for all $k \in \{0, \ldots, j-1\}\backslash\{i\}$. ($G$ may have parallel edges.) If $|E(G)| \leqslant m$, we are done.

Otherwise, G contains a circuit $(\{v_1, \ldots, v_k, v_{k+1} = v_1\}, \{\{v_i, v_{i+1}\} : i = 1, \ldots, k\})$. For each $i \in \{1, \ldots, k\}$ there is a $c_i \in C$ with $0 < g(c_i, v_i) < 1$ and $0 < g(c_i, v_{i+1}) < 1$ (here $v_{k+1} := v_1$). $c_1, \ldots, c_k$ are pairwise distinct. Hence for a sufficiently small $\varepsilon > 0$, we have that g' and g'' are feasible fractional partitions, where $g'(c_i, v_i) := g(c_i, v_i) - \varepsilon/\text{size}(c_i)$, $g'(c_i, v_{i+1}) := g(c_i, v_{i+1}) + \varepsilon/\text{size}(c_i)$, $g''(c_i, v_i) := g(c_i, v_i) + \varepsilon/\text{size}(c_i)$, $g''(c_i, v_{i+1}) := g(c_i, v_{i+1}) - \varepsilon/\text{size}(c_i)$ $(i = 1, \ldots, k)$ and $g'(c) := g''(c) := g(c)$ for $c \in C\backslash\{c_1, \ldots, c_k\}$.

The arithmetic mean of the objective function values of g' and g'' is precisely that of g, implying that g' and g'' are also optimum. If we choose ε as large as possible, $\Phi(g')$ or $\Phi(g'')$ is strictly smaller than $\Phi(g)$. This contradicts the choice of g. □

A similar statement has been proved in [16]. The above proof directly yields a linear-time algorithm (for constant m) which, given an optimum fractional partition g, finds another one g' which is integral except for at most m objects. By rounding the values for these few objects we get a partition which is "almost feasible and almost optimum". Whenever the number of elements of C is large enough and the size of every single element is small with respect to the total size, this will suffice for practical purposes. In particular, this is the case in VLSI placement. We recall the following result by Shmoys and Tardos [18]:

Theorem 2.2. *For any instance of the* Generalized Assignment Problem *and any optimum fractional partition, we can compute in polynomial time an integral partition f with the same cost, and satisfying $\sum\{\text{size}(c,j) : f(c) = j\} \leqslant \kappa_j + \max\{\text{size}(c,j) : c \in C\}$ for $j = 0, 1, \ldots, m$.*

Indeed, such a partition can be computed in constant time (for constant m) when starting with a fractional partition as in Proposition 2.1. If we ask for a feasible integral partition, we can use the following observation. For $X \subseteq C$ we write $\text{size}(X) := \sum_{c\in X} \text{size}(c)$.

Proposition 2.3. *For an instance of the* Multiple Assignment Problem *let $C' \subseteq C$ be the subset of circuits c with $\text{size}(c) \geqslant (\sum_{i=0}^{m} \kappa_i - \text{size}(C))/(m+1) =: M$; let $k := |C'|$. For constant k and m we can get a feasible partition*

in $O(n \log^2 n)$ *time, such that the objective function value differs from the optimum by at most* $mr_{\max}$, *where* $r_{\max} = \max\{(r(c,i) - r(c,j))M/\text{size}(c) : c \in C \setminus C',\ i \neq j\}$.

Proof. Let $C' \subseteq C$ be as above. We try all possible partitions for C'. For each of the at most $(m+1)^k$ possibilities we consider the remaining problem (on the set $C'' := C \setminus C'$ with remaining capacities, say, κ'_i $(i = 0, \ldots, m)$). The remaining capacities of course depend on the partition of C'. But in any case, we can decrease the capacities to $\kappa''_i := \max\{0, \kappa'_i - M\}$ $(i = 0, \ldots, m)$ and still have $\text{size}(C'') \leqslant \sum_{i=0}^{m} \kappa''_i$. The reduction of the capacities increases the fractional optimum by at most $mr_{\max}$.

So there is a feasible fractional partition of C'' with respect to capacities κ''_i, $i = 0, \ldots, m$. We find an optimum one in $O(n \log^2 n)$ time by the above reduction to the Minimum Cost Flow Problem and the algorithm of [19]. Indeed, by Proposition 2.1 we may assume that our optimum fractional partition g of C'' is integral on $C'' \setminus C'''$, where $|C'''| \leqslant m$.

Thus, we can apply Theorem 2.2 and obtain a feasible partition f of C'' with respect to capacities κ' in constant time, at no extra cost. □

The value of $r_{\max}$ can often also be considered as constant. In this case, we have an absolute approximation algorithm.

However, the running time is still too high for large practical instances where n is in the millions. Also, the above-mentioned weakly polynomial $O(n + \log R)$ bound of the bipartite network flow algorithm by [1] is often not attractive. One important example is VLSI placement. Fortunately, the instances arising there have a special structure which can be exploited to get a strongly polynomial linear-time algorithm. This will be described in the rest of this paper. We shall use the above construction of the Minimum Cost Flow Problem again (in a slightly different form) in the proof of Theorem 5.3.

3. VLSI placement and quadrisection

Since VLSI placement was the main motivation for this research we briefly discuss this application. For more details on VLSI layout see, e.g. [15] or [24].

In VLSI placement one has a set of objects (called cells, components, or circuits) which have to be placed within some given chip area, without any overlaps. Each object has pins which then, in the routing phase, have to be connected in a certain way. To simplify the routing task and for timing reasons it is good if objects that must be connected are close together. Therefore, a main objective in placement is to minimize some estimation of the interconnect length.

Either placing the objects without overlaps or placing them with minimum estimated interconnect is usually not difficult, but the combination is very hard. So far no efficient algorithm with reasonable performance guarantee exists, the best known (polylogarithmic) approximation algorithm is due to Even et al. [9]. Since the problem is of outstanding practical importance, one has to use heuristics.

Most state-of-the-art placement algorithms for large instances (there are chips with several million objects today) proceed as follows: they successively partition the chip area to smaller and smaller regions, and in each step distribute the objects to be placed to the regions. Of course, no region should contain more objects than fit into it.

The question is what strategy should be used for partitioning the objects of one region to its subregions. For a long time, a min-cut objective was popular, minimizing the number of connections between different regions (see [2] for a survey). Tsay et al. [21] proposed a different method: take the placement with minimum estimated interconnect length (however, with usually many overlaps) as a starting point and try to modify it as little as possible in order to get a feasible partition (i.e. meeting capacity constraints). This problem can be easily solved almost optimally for bipartitioning (see Section 4). For quadrisection, [21] suggest heuristics. In this paper, we show how to solve the Quadrisection Problem almost optimally in linear time. This algorithm is a main component of the Bonn placement algorithm (see [22,5]), which has been successfully used by IBM for more than hundred of the most complex industrial chip designs in the last years.

Figs. 14 and 15 (in the Appendix) show two real-world examples. While the large blocks (gray) have been fixed, the others are placed such that the sum of the squared wirelength estimations is minimum. The colors then show the optimum quadrisection. The common structure of the color maps (which we will call *American map*) is no coincidence as we shall see in Section 7.

This generic placement approach leads to the Quadrisection Problem which is the subject of this paper. We first describe the instances. We are given a finite set C, coordinates $(x(c), y(c)) \in \mathbb{R}^2$ and the size, a positive number $\text{size}(c)$ for each $c \in C$.

Moreover, let $x_1, y_1 \in \mathbb{R}$ be two coordinates defining four regions $R_0 = \{(x, y) \in \mathbb{R}^2 : x \leqslant x_1,\ y \leqslant y_1\}$, $R_1 = \{(x, y) \in \mathbb{R}^2 : x \geqslant x_1,\ y \leqslant y_1\}$, $R_2 = \{(x, y) \in \mathbb{R}^2 : x \geqslant x_1,\ y \geqslant y_1\}$, $R_3 = \{(x, y) \in \mathbb{R}^2 : x \leqslant x_1,\ y \geqslant y_1\}$ (see Fig. 1). Finally, we are given non-negative capacities $\kappa_0, \kappa_1, \kappa_2, \kappa_3 \geqslant 0$ of the four regions. We assume $\text{size}(C) \leqslant \sum_{i=0}^{3} \kappa_i$.

Throughout this paper, we assume such an instance be given. With this we can define the fractional relaxation of the Quadrisection Problem formally.

Definition 3.1. A *feasible partition* of C is a mapping $f : C \to \{0, 1, 2, 3\}$ with

$$\text{size}(\{c \in C : f(c) = i\}) \leqslant \kappa_i$$

for all $i \in \{0, 1, 2, 3\}$.

A *feasible fractional partition* of C is a mapping $g : C \times \{0, 1, 2, 3\} \to [0, 1]$ with $\sum_{i=0}^{3} g(c, i) = 1$ for all $c \in C$ and

$$\sum_{c \in C} g(c, i)\text{size}(c) \leqslant \kappa_i$$

for all $i \in \{0, 1, 2, 3\}$.

The *total movement* of a feasible fractional partition g is defined as

$$\sum_{c \in C} \text{size}(c) \sum_{i=0}^{3} g(c, i)\ d((x(c), y(c)), R_i),$$

where $d((x, y), R_i) := \min_{(x', y') \in R_i} (|x - x'| + |y - y'|)$ denotes the ℓ_1-distance. A feasible fractional partition is called *optimum* if its total movement is minimum.

The ℓ_1-distance is the natural measure in VLSI design where only horizontal and vertical wires are allowed. One might ask why the contribution of a component's movement to the objective function is proportional to its size. Although this seems to make sense in the VLSI placement application, there might be other applications where this is not the case. However, one can also use other weights than the sizes in the objective function.

In general, one can find in linear time a feasible fractional partition minimizing

$$\sum_{c \in C} \sum_{i=0}^{3} g(c, i)\ r(c, i), \tag{*}$$

where $r : C \times \{0, 1, 2, 3\} \to \mathbb{R}$ is an arbitrary cost function satisfying $r(c, 0) + r(c, 2) = r(c, 1) + r(c, 3)$ for all $c \in C$. For example, this includes the weighted sum of ℓ_1-distances as well as of squared Euclidean distances.

To reduce the above optimization problem with objective function $(*)$ to the Quadrisection Problem, let $r : C \times \{0, 1, 2, 3\} \to \mathbb{R}$ with $r(c, 0) + r(c, 2) = r(c, 1) + r(c, 3)$ for all $c \in C$. For each $c \in C$ we compute $\min\{r(c, i) : i \in \{0, 1, 2, 3\}\}$; suppose the minimum is attained in $i_c \in \{0, 1, 2, 3\}$. Let now j_c and k_c be the horizontally and vertically adjacent region, respectively, i.e.

$$j_c = \begin{cases} 1 & \text{if } i_c = 0, \\ 0 & \text{if } i_c = 1, \\ 3 & \text{if } i_c = 2, \\ 2 & \text{if } i_c = 3 \end{cases} \quad \text{and} \quad k_c = \begin{cases} 3 & \text{if } i_c = 0, \\ 2 & \text{if } i_c = 1, \\ 1 & \text{if } i_c = 2, \\ 0 & \text{if } i_c = 3. \end{cases}$$

We then compute a point $(x'(c), y'(c)) \in R_{i_c}$ by

$$|x_1 - x'(c)| = \frac{r(c, j_c) - r(c, i_c)}{\text{size}(c)}$$

and

$$|y_1 - y'(c)| = \frac{r(c, k_c) - r(c, i_c)}{\text{size}(c)}.$$

If we now solve the Quadrisection Problem with the coordinates $(x'(c), y'(c))$ (minimizing the total movement), this is obviously equivalent to minimizing $(*)$.

We henceforth consider the fractional relaxation of the Quadrisection Problem only: find an optimum fractional partition. Of course, eventually we are interested in integral solutions (i.e. feasible partitions). Therefore, Propositions 2.1 and 2.3 are essential.

In the rest of this paper, we assume C, size(c) and $(x(c), y(c))$ for $c \in C$, x_1, y_1 and $\kappa_0, \kappa_1, \kappa_2, \kappa_3$ to be given. Without loss of generality, we make the following assumptions:

- $\kappa_0, \kappa_1, \kappa_2, \kappa_3 > 0$.
- $\text{size}(C) = \sum_{i=0}^{3} \kappa_i$.

To justify the first assumption, suppose, say $\kappa_0 = 0$. Then note that adding an element c with $\text{size}(c) = 1$, $x(c) < x(c')$ and $y(c) < y(c')$ for all $c' \in C$ and increasing κ_0 by one yields an equivalent instance.

To justify the second assumption, note that $\text{size}(C) > \sum_{i=0}^{3} \kappa_i$ implies infeasibility of the instance. If $\delta := \sum_{i=0}^{3} \kappa_i - \text{size}(C) > 0$, we may add an element c with $\text{size}(c) = \delta$, $x(c) = x_1$ and $y(c) = y_1$. Evidently, any optimum solution of the extended instance yields an optimum solution for the original instance.

4. Bipartitioning and weighted median

It is instructive to consider first the bipartitioning case, i.e. two of the four capacities are zero. In this case, optimal fractional partitions have a very simple form:

Proposition 4.1. *Let $\kappa_2 = \kappa_3 = 0$. Then there exists a number $\bar{x}$ and an optimum fractional partition g of C with $x(c) < \bar{x} \Rightarrow g(c, 0) = 1$ and $x(c) > \bar{x} \Rightarrow g(c, 1) = 1$ for all $c \in C$.*

Proof. If the partition $g(c, 0) = 1$ for $c \in C$ with $x(c) \leqslant x_1$ and $g(c, 1) = 1$ otherwise is feasible, then it is optimum (due to total movement zero) and proves the assertion ($\bar{x} := x_1$). Otherwise, let w.l.o.g. $\text{size}(\{c \in C : x(c) \leqslant x_1\}) > \kappa_0$. Let $C = \{c_1, \ldots, c_n\}$ with $x(c_1) \leqslant \cdots \leqslant x(c_n)$, and let

$$k := \min \left\{ j \in \{1, \ldots, n\} : \sum_{i=1}^{j} \text{size}(c_i) > \kappa_0 \right\}.$$

By setting $\bar{x} := x(c_k)$, $g(c_i, 0) = 1$ for $i = 1, \ldots, k-1$,

$$g(c_k, 0) = \frac{\kappa_0 - \sum_{i=1}^{k-1} \text{size}(c_i)}{\text{size}(c_k)}$$

and $g(c_i, 0) = 0$ for $i = k+1, \ldots, n$, one obviously obtains an optimum fractional partition. □

An analogous observation for the fractional Knapsack Problem was made by Dantzig [8]; the same idea appears already in Tolstoĭ's early work on the transportation problem [20]. It follows that an optimum solution can be found in $O(n \log n)$ time by sorting. In fact, a linear running time is possible: just observe that Proposition 4.1 reduces the problem to a weighted median search.

Definition 4.2. Let $n \in \mathbb{N}$, $z_1, \ldots, z_n \in \mathbb{R}$ and $w_1, \ldots, w_n \in \mathbb{R}_+$ and $0 < w^* \leqslant \sum_{i=1}^n w_i$. The $(w_1, \ldots, w_n; w^*)$-*weighted median* w.r.t. $(z_1, \ldots, z_n)$ is the unique number z^* with

$$\sum_{\substack{i \in \{1,\ldots,n\} \\ z_i < z^*}} w_i < w^* \leqslant \sum_{\substack{i \in \{1,\ldots,n\} \\ z_i \leqslant z^*}} w_i .$$

By the *non-weighted median* w.r.t. $(z_1, \ldots, z_n)$ we mean the $(1, \ldots, 1; n/2)$-weighted median w.r.t. $(z_1, \ldots, z_n)$.

In the case of Proposition 4.1 (let $C = \{c_1, \ldots, c_n\}$), we set $w_i := \text{size}(c_i)$, $z_i := x(c_i)$ $(i = 1, \ldots, n)$ and

$$w^* := \min\{\kappa_0, \text{size}(\{c \in C : x(c) \leqslant x_1\})\} + \max\{0, \text{size}(\{c \in C : x(c) > x_1\}) - \kappa_1\}.$$

Then the weighted median z^* is a $\bar{x}$ according to Proposition 4.1. Please observe that the other bipartitioning cases (when two other capacities are zero) can be treated similarly.

It is well-known that the weighted median can be determined in $O(n)$ time; see e.g. [14]. The algorithm is essentially due to Blum et al. [3].

5. Maps

In the following three sections, we shall develop a theorem on the structure of optimum fractional partitions, similar to Proposition 4.1 in the special case of bipartitioning. Starting point is the following basic idea: we partition the plane into four "countries" L_0, L_1, L_2, L_3 such that the fractional partition g shall satisfy $g(c, i) > 0 \Rightarrow (x(c), y(c)) \in L_i$. The countries shall be closed sets intersecting only in the boundaries. By $\overset{\circ}{S}$ we denote the interior of a set $S \subseteq \mathbb{R}^2$.

Definition 5.1. A *map* is a quadruple (L_0, L_1, L_2, L_3) of closed sets in $\mathbb{R}^2$ with $L_0 \cup L_1 \cup L_2 \cup L_3 = \mathbb{R}^2$ and $\overset{\circ}{L}_i \cap L_j = \emptyset$ for all $i \neq j$.

A map defines a partition of C except for elements on the boundary of two countries.

Definition 5.2. Given a fractional partition g of C and a map $\mathcal{L} = (L_0, L_1, L_2, L_3)$, g is *consistent* with $\mathcal{L}$ if for all $c \in C$ and $i \in \{0, 1, 2, 3\}$ with $g(c, i) > 0$ we have $(x(c), y(c)) \in L_i$.

A map is called *feasible* if there exists a feasible fractional partition of C consistent with it.

So, now, we look for a feasible map. Thus, a simple criterion for the feasibility of maps will be useful.

Theorem 5.3. *Let $\mathcal{L} = (L_0, L_1, L_2, L_3)$ be a map. $\mathcal{L}$ is feasible if and only if*

$$\text{size}\left(\left\{c \in C : (x(c), y(c)) \notin \bigcup_{i \in \{0,1,2,3\} \setminus I} L_i\right\}\right) \leqslant \sum_{i \in I} \kappa_i$$

for all $I \subseteq \{0, 1, 2, 3\}$. If $\mathcal{L}$ is feasible, then a feasible fractional partition of C consistent with $\mathcal{L}$ can be found in linear time (assuming an oracle deciding in which countries L_i a given point lies).

Proof. The necessity of the condition is obvious. To prove sufficiency we define an auxiliary digraph G by $V(G) := C \dot{\cup} \{0, 1, 2, 3, s, t\}$ and

$$E(G) := \{(s, c) : c \in C\} \cup \{(c, i) : (x(c), y(c)) \in L_i\} \cup \{(i, t) : i \in \{0, 1, 2, 3\}\},$$

and capacities $u((s, c)) := \text{size}(c)$, $u((i, t)) := \kappa_i$ and $u((c, i)) := \infty$ for $c \in C$ and $i \in \{0, 1, 2, 3\}$. Let now $E(X, V(G) \setminus X)$ be an s–t-cut, say $X = \{s\} \cup C_1 \cup I_1$ with $C_1 \subseteq C$ and $I_1 \subseteq \{0, 1, 2, 3\}$. If this s–t-cut has finite

capacity, then $(x(c), y(c)) \notin L_i$ for $c \in C_1$ and $i \in \{0, 1, 2, 3\} \setminus I_1$. Hence, the capacity of the s–t-cut is at least

$$\text{size}\left(\left\{c \in C : (x(c), y(c)) \in \bigcup_{i \in \{0,1,2,3\} \setminus I_1} L_i\right\}\right) + \sum_{i \in I_1} \kappa_i,$$

and therefore at least size(C). Then, by the Max-Flow-Min-Cut Theorem, there is an s–t-flow f with value size(C). By setting $g(c, i) := f((c, i))/\text{size}(c)$ this yields a feasible fractional partition g consistent with $\mathscr{L}$.

To see the linear running time, observe that for elements $c, c' \in C$ with $\{i : (x(c), y(c)) \in L_i\} = \{i : (x(c'), y(c')) \in L_i\}$ we need only one vertex. By this reduction we end up with at most 21 vertices, and a maximum s–t-flow can be found in constant time. Determining g in the end takes linear time, and so does the initial construction of the graph. □

Results of Boros and Hwang [4] imply that one can restrict attention to maps whose countries are all convex, even in a more general context. In our case it will turn out that we need to consider only maps of a very special type:

Definition 5.4. For numbers $z, w_1, w_2 \in \mathbb{R}$ let $\mathscr{L}(z, w_1, w_2) := (L_0, L_1, L_2, L_3)$, where

$$L_0 = \left\{(x, y) \in \mathbb{R}^2 : x + y \leqslant z,\ x \leqslant \frac{z - w_1}{2},\ y \leqslant \frac{z + w_2}{2}\right\},$$
$$L_1 = \left\{(x, y) \in \mathbb{R}^2 : x \geqslant \frac{z - w_1}{2},\ y \leqslant \frac{z + w_1}{2}\right\},$$
$$L_2 = \left\{(x, y) \in \mathbb{R}^2 : x + y \geqslant z,\ x \geqslant \frac{z - w_2}{2},\ y \geqslant \frac{z + w_1}{2}\right\},$$
$$L_3 = \left\{(x, y) \in \mathbb{R}^2 : x \leqslant \frac{z - w_2}{2},\ y \geqslant \frac{z + w_2}{2}\right\}.$$

For numbers $w, z_1, z_2 \in \mathbb{R}$ let $\mathscr{L}'(w, z_1, z_2) := (L_0, L_1, L_2, L_3)$, where

$$L_0 = \left\{(x, y) \in \mathbb{R}^2 : x \leqslant \frac{z_1 - w}{2},\ y \leqslant \frac{z_1 + w}{2}\right\},$$
$$L_1 = \left\{(x, y) \in \mathbb{R}^2 : y - x \leqslant w,\ x \geqslant \frac{z_1 - w}{2},\ y \leqslant \frac{z_2 + w}{2}\right\},$$
$$L_2 = \left\{(x, y) \in \mathbb{R}^2 : x \geqslant \frac{z_2 - w}{2},\ y \geqslant \frac{z_2 + w}{2}\right\},$$
$$L_3 = \left\{(x, y) \in \mathbb{R}^2 : y - x \geqslant w,\ x \leqslant \frac{z_2 - w}{2},\ y \geqslant \frac{z_1 + w}{2}\right\}.$$

A map $\mathscr{L}$ is called *American*, if there are numbers z, w_1, w_2 with $w_1 \leqslant w_2$ and $\mathscr{L} = \mathscr{L}(z, w_1, w_2)$, or there are numbers w, z_1, z_2 with $z_1 \leqslant z_2$ and $\mathscr{L} = \mathscr{L}'(w, z_1, z_2)$.

Fig. 2 shows a sketch of three American maps. The name alludes to the straight borders between American states.

As usual the letters x and y describe horizontal and vertical coordinates. The letters z and w are used to determine a diagonal from the lower right to the upper left, respectively from the lower left to the upper right (more precisely a point set $\{(x, y) : x + y = z\}$ or $\{(x, y) : y - x = w\}$). The endpoints of segments of such lines are denoted by w_1, w_2 or z_1, z_2, i.e. a diagonal line segment by $\{(x, y) : x + y = z,\ w_1 \leqslant y - x \leqslant w_2\}$ or $\{(x, y) : y - x = w,\ z_1 \leqslant x + y \leqslant z_2\}$. In the following sections the letters x, y, z, w will be used only in this context.

In the following sections, we shall exhibit the crucial role of American maps: there always exists an optimum fractional partition which is consistent with an American map (this will be Theorem 7.3). We remark that this theorem has a simpler proof (see the end of Section 7), but the linear-time algorithm of Section 8 is based on the following.

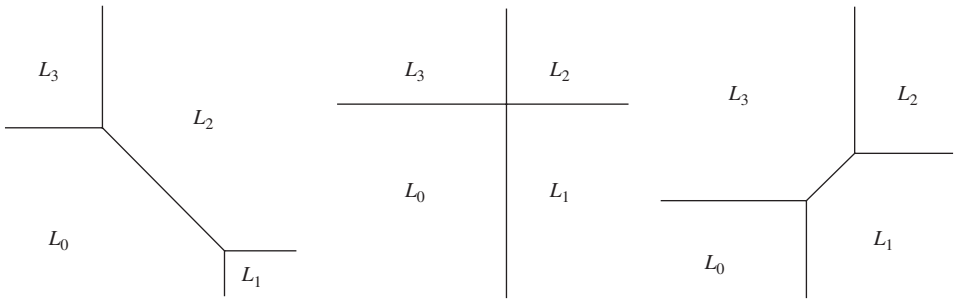

Fig. 2.

6. Existence of feasible American maps

Now, we show that there always exists an American map and a feasible fractional partition consistent with it. In Section 7, we then show that such a fractional partition must be optimum.

For $z \in \mathbb{R}$ let

$$w_1(z) := \inf\left\{w : \text{size}\left(\left\{c \in C : x(c) \geqslant \frac{z-w}{2},\ \ y(c) \leqslant \frac{z+w}{2}\right\}\right) \geqslant \kappa_1\right\},$$
$$w_2(z) := \sup\left\{w : \text{size}\left(\left\{c \in C : x(c) \leqslant \frac{z-w}{2},\ \ y(c) \geqslant \frac{z+w}{2}\right\}\right) \geqslant \kappa_3\right\}$$

and

$$(P_0(z), P_1(z), P_2(z), P_3(z)) := \mathscr{L}(z, w_1(z), w_2(z)).$$

For $w \in \mathbb{R}$ let

$$z_1(w) := \inf\left\{z : \text{size}\left(\left\{c \in C : x(c) \leqslant \frac{z-w}{2},\ \ y(c) \leqslant \frac{z+w}{2}\right\}\right) \geqslant \kappa_0\right\},$$
$$z_2(w) := \sup\left\{z : \text{size}\left(\left\{c \in C : x(c) \geqslant \frac{z-w}{2},\ \ y(c) \geqslant \frac{z+w}{2}\right\}\right) \geqslant \kappa_2\right\}$$

and

$$(Q_0(z), Q_1(z), Q_2(z), Q_3(z)) := \mathscr{L}'(w, z_1(w), z_2(w)).$$

Finally, let

$$\bar{z} := \inf\left\{ z : \text{size}\left(\left\{c \in C : (x(c), y(c)) \in \bigcup_{i \in I} P_i(z)\right\}\right) \geqslant \sum_{i \in I} \kappa_i \ \text{ for all } I \text{ with } 0 \in I \subseteq \{0, 1, 3\}\right\}$$

and

$$\bar{w} := \inf\left\{ w : \text{size}\left(\left\{c \in C : (x(c), y(c)) \in \bigcup_{i \in I} Q_i(w)\right\}\right) \geqslant \sum_{i \in I} \kappa_i \ \text{ for all } I \text{ with } 1 \in I \subseteq \{0, 1, 2\}\right\}.$$

$\mathscr{L}(\bar{z}, w_1(\bar{z}), w_2(\bar{z}))$ and $\mathscr{L}'(\bar{w}, z_1(\bar{w}), z_2(\bar{w}))$ are not necessarily maps; Fig. 3 illustrates $\mathscr{L}(\bar{z}, w_1(\bar{z}), w_2(\bar{z}))$ in the cases $w_1(\bar{z}) < w_2(\bar{z})$ and $w_2(\bar{z}) < w_1(\bar{z})$.

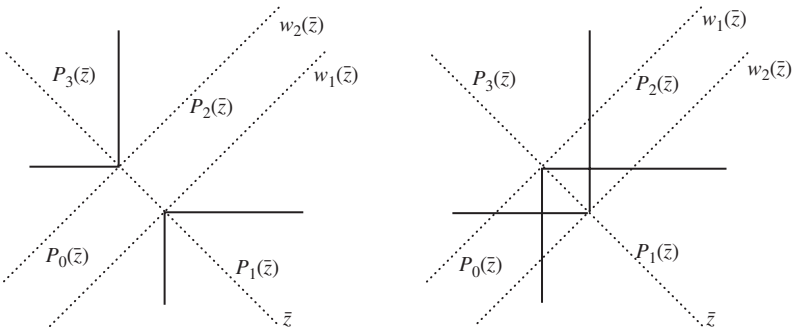

Fig. 3.

The idea of the above definitions is as follows. Because of $\text{size}(C) = \sum_{i=0}^{3} \kappa_i$ and Theorem 5.3 a map $(P_0(z), P_1(z), P_2(z), P_3(z))$ is feasible if and only if

$$\text{size}\left(\left\{c \in C : (x(c), y(c)) \in \bigcup_{i \in I} P_i(z)\right\}\right) \geqslant \sum_{i \in I} \kappa_i \quad \text{for all } I \subseteq \{0, 1, 2, 3\}.$$

For constant z this condition for $\mathscr{L}(z, w_1(z), w_2(z))$ and $I = \{1\}$ and $I = \{3\}$ is ensured by the choice of $w_1(z)$ and $w_2(z)$, for all I with $0 \in I \subseteq \{0, 1, 3\}$ by the choice of $\bar z$. By choosing these numbers minimally, respectively maximally, the condition is also ensured for the other sets I, unless $w_1(\bar z) > w_2(\bar z)$. This will be shown in Lemma 6.3. However, $w_1(\bar z) > w_2(\bar z)$ remains possible. But in this case we must have $z_1(\bar w) \leqslant z_2(\bar w)$, and we have a feasible American map of the second type. This will be Lemma 6.2.

Instead of inf and sup we can always write min and max, because C is finite.

The existence of $\bar z$ and $\bar w$ follows from the fact that $\max\{x(c) : c \in C\} + \max\{y(c) : c \in C\}$ and $\max\{y(c) : c \in C\} - \min\{x(c) : c \in C\}$, respectively, is always contained in the set over which the infimum is taken. This, in turn, is a consequence of the following lemma:

Lemma 6.1. *Let* $x, y \in \mathbb{R}$.

If $\text{size}(\{c \in C : x(c) > x,\ y(c) \leqslant y\}) < \kappa_1$, $\text{size}(\{c \in C : x(c) \leqslant x,\ y(c) > y\}) < \kappa_3$ *and* $\text{size}(\{c \in C : x(c) \leqslant x \vee y(c) \leqslant y\}) \geqslant \kappa_0 + \kappa_1 + \kappa_3$, *then* $\bar z \leqslant x + y$.

If $\text{size}(\{c \in C : x(c) < x,\ y(c) \leqslant y\}) < \kappa_0$, $\text{size}(\{c \in C : x(c) \geqslant x,\ y(c) > y\}) < \kappa_2$ *and* $\text{size}(\{c \in C : x(c) \geqslant x \vee y(c) \leqslant y\}) \geqslant \kappa_0 + \kappa_1 + \kappa_2$, *then* $\bar w \leqslant y - x$.

Proof. By symmetry it suffices to prove the first assertion (compare Fig. 4). Let $z := x + y$. From the definitions of $w_1(z)$ and $w_2(z)$ one can deduce that $w_2(z) \leqslant y - x \leqslant w_1(z)$. Hence,

$$\text{size}(\{c \in C : (x(c), y(c)) \in P_0(z) \cup P_1(z) \cup P_3(z)\}) \geqslant \kappa_0 + \kappa_1 + \kappa_3.$$

Moreover,

$$\begin{aligned}
&\text{size}(\{c \in C : (x(c), y(c)) \in P_0(z)\}) \\
&\quad \geqslant \text{size}(\{c \in C : x(c) \leqslant x,\ y(c) \leqslant y\}) - (\kappa_1 - \text{size}(\{c \in C : x(c) > x,\ y(c) \leqslant y\})) \\
&\qquad - (\kappa_3 - \text{size}(\{c \in C : x(c) \leqslant x,\ y(c) > y\})) \\
&\quad = \text{size}(\{c \in C : x(c) \leqslant x \vee y(c) \leqslant y\}) - \kappa_1 - \kappa_3 \\
&\quad \geqslant \kappa_0.
\end{aligned}$$

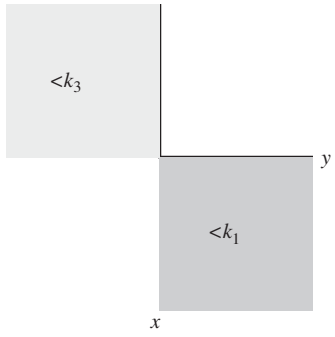

Fig. 4.

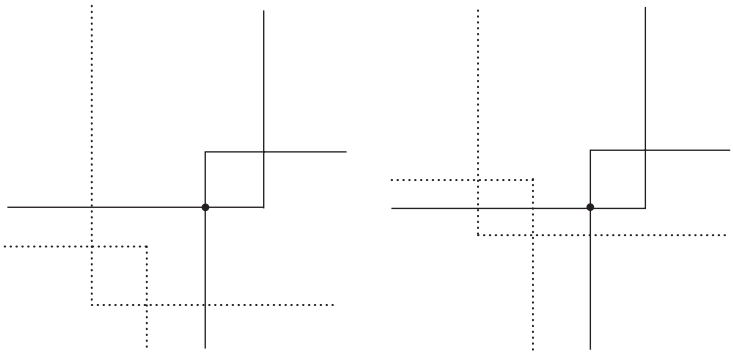

Fig. 5.

Finally, we have

$$\begin{aligned}
&\text{size}(\{c \in C : (x(c), y(c)) \in P_0(z) \cup P_1(z)\})\\
&\quad \geqslant \text{size}(\{c \in C : y(c) \leqslant y\}) - (\kappa_3 - \text{size}(\{c \in C : x(c) \leqslant x,\ y(c) > y\}))\\
&\quad = \text{size}(\{c \in C : x(c) \leqslant x \vee y(c) \leqslant y\}) - \kappa_3\\
&\quad \geqslant \kappa_0 + \kappa_1
\end{aligned}$$

and analogously

$$\text{size}(\{c \in C : (x(c), y(c)) \in P_0(z) \cup P_3(z)\}) \geqslant \kappa_0 + \kappa_3,$$

implying the claim $\bar{z} \leqslant z$. □

The rest of this section consists of the proof that at least one of $(P_0(\bar{z}), P_1(\bar{z}), P_2(\bar{z}), P_3(\bar{z}))$ and $(Q_0(\bar{w}), Q_1(\bar{w}), Q_2(\bar{w}), Q_3(\bar{w}))$ is a feasible American map. First, we show that at least one of them is a map at all.

Lemma 6.2. *At least one of the two statements $w_1(\bar{z}) \leqslant w_2(\bar{z})$ and $z_1(\bar{w}) \leqslant z_2(\bar{w})$ is true.*

Proof. We show that the assumption $w_1(\bar{z}) > w_2(\bar{z})$ and $z_1(\bar{w}) > z_2(\bar{w})$ leads to a contradiction.

First, we observe that this assumption implies $(z_1(\bar{w}) - \bar{w})/2 \leqslant (\bar{z} - w_1(\bar{z}))/2$, because otherwise $P_0 \subseteq \overset{\circ}{Q}_0$ or $Q_1 \subseteq \overset{\circ}{P}_1$; but this is impossible due to the minimal choice of $z_1(\bar{w})$ and the definition of $\bar{z}$, and due to the minimal choice of $w_1(\bar{z})$ and the definition of $\bar{w}$.

Now, we distinguish two cases, depending on whether $(\bar{z} + w_2(\bar{z}))/2 \geqslant (z_2(\bar{w}) + \bar{w})/2$ (Fig. 5) or $(\bar{z} + w_2(\bar{z}))/2 < (z_2(\bar{w}) + \bar{w})/2$ (Fig. 6). In these figures, solid lines bound $P_1(\bar{z})$ and $P_3(\bar{z})$, and dotted lines bound $Q_0(\bar{w})$ and $Q_2(\bar{w})$.

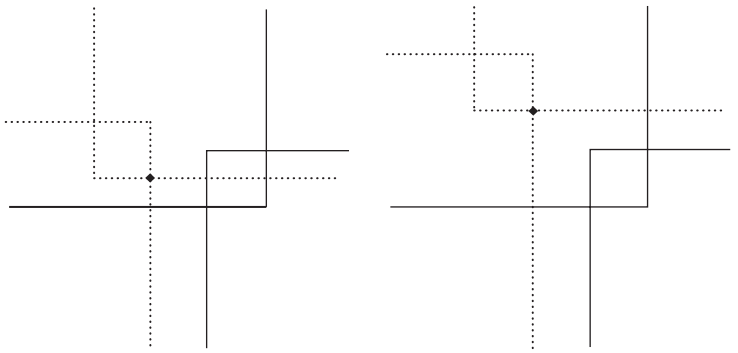

Fig. 6.

In the first case, let $x := (\bar{z} - w_1(\bar{z}))/2$ and $y := (\bar{z} + w_2(\bar{z}))/2$. Then (x, y) (the small bullet in Fig. 5) satisfies the conditions of Lemma 6.1 (first part), because

$$\begin{aligned}\text{size}(\{c \in C : x(c) > x,\ \ y(c) \leqslant y\}) &\leqslant \text{size}\left(\left\{c \in C : (x(c), y(c)) \in \overset{\circ}{P_1}(\bar{z})\right\}\right)\\ &< \kappa_1,\end{aligned}$$

$$\begin{aligned}\text{size}(\{c \in C : x(c) \leqslant x,\ \ y(c) > y\}) &\leqslant \text{size}\left(\left\{c \in C : (x(c), y(c)) \in \overset{\circ}{P_3}(\bar{z})\right\}\right)\\ &< \kappa_3\end{aligned}$$

and

$$\begin{aligned}\text{size}(\{c \in C : x(c) \leqslant x \vee y(c) \leqslant y\}) &\geqslant \text{size}\left(\left\{c \in C : (x(c), y(c)) \notin \overset{\circ}{Q_2}(\bar{w})\right\}\right)\\ &> \kappa_0 + \kappa_1 + \kappa_3.\end{aligned}$$

Applying Lemma 6.1 then results in a contradiction with $\bar{z} > x + y$.

In the second case, let $x := (z_1(\bar{w}) - \bar{w})/2$ and $y := (z_2(\bar{w}) + \bar{w})/2$. Then (x, y) (the small bullet in Fig. 6) satisfies the conditions of Lemma 6.1 (second part), because

$$\begin{aligned}\text{size}(\{c \in C : x(c) < x,\ \ y(c) \leqslant y\}) &\leqslant \text{size}\left(\left\{c \in C : (x(c), y(c)) \in \overset{\circ}{Q_0}(\bar{z})\right\}\right)\\ &< \kappa_0,\end{aligned}$$

$$\begin{aligned}\text{size}(\{c \in C : x(c) \geqslant x,\ \ y(c) > y\}) &\leqslant \text{size}\left(\left\{c \in C : (x(c), y(c)) \in \overset{\circ}{Q_2}(\bar{z})\right\}\right)\\ &< \kappa_2\end{aligned}$$

and

$$\begin{aligned}\text{size}(\{c \in C : x(c) \geqslant x \vee y(c) \leqslant y\}) &\geqslant \text{size}\left(\left\{c \in C : (x(c), y(c)) \notin \overset{\circ}{P_3}(\bar{w})\right\}\right)\\ &> \kappa_0 + \kappa_1 + \kappa_2.\end{aligned}$$

Applying Lemma 6.1 then results in a contradiction with $\bar{w} > y - x$. □

Hence, at least one of $(P_0(\bar{z}), P_1(\bar{z}), P_2(\bar{z}), P_3(\bar{z}))$ and $(Q_0(\bar{w}), Q_1(\bar{w}), Q_2(\bar{w}), Q_3(\bar{w}))$ is a map. Now, we show that this map satisfies the requirements of Theorem 5.3, i.e. is feasible.

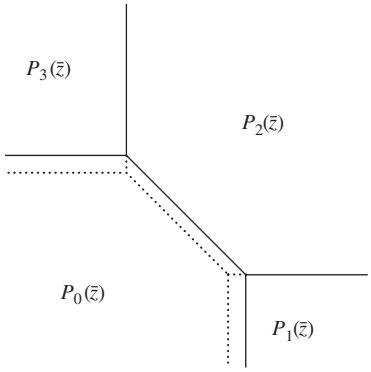

Fig. 7.

Lemma 6.3. *If $w_1(\bar{z}) \leqslant w_2(\bar{z})$, then we have for each $I \subseteq \{0, 1, 2, 3\}$:*

$$\text{size}\left(\left\{c \in C : (x(c), y(c)) \notin \bigcup_{i \in \{0,1,2,3\}\setminus I} P_i(\bar{z})\right\}\right) \leqslant \sum_{i \in I} \kappa_i.$$

If $z_1(\bar{w}) \leqslant z_2(\bar{w})$, then we have for each $I \subseteq \{0, 1, 2, 3\}$:

$$\text{size}\left(\left\{c \in C : (x(c), y(c)) \notin \bigcup_{i \in \{0,1,2,3\}\setminus I} Q_i(\bar{w})\right\}\right) \leqslant \sum_{i \in I} \kappa_i.$$

Proof. By symmetry, it suffices to prove the first statement. Hence, we assume $w_1(\bar{z}) \leqslant w_2(\bar{z})$.

For $I = \emptyset$ or $I = \{0, 1, 2, 3\}$ the assertion is trivial (we have equality). For $I = \{0, 2, 3\}$ and $I = \{0, 1, 2\}$ the assertion follows directly from the choice of $w_1(\bar{z})$ and $w_2(\bar{z})$. For $I = \{1, 2, 3\}$, $I = \{2, 3\}$, $I = \{1, 2\}$ and $I = \{2\}$ the assertion follows directly from the choice of $\bar{z}$.

Moreover, note that due to the choice of $w_1(\bar{z})$ and $w_2(\bar{z})$ we have

$$\text{size}\left(\left\{c \in C : (x(c), y(c)) \in \overset{\circ}{P_1}(\bar{z})\right\}\right) < \kappa_1 \tag{1}$$

and

$$\text{size}\left(\left\{c \in C : (x(c), y(c)) \in \overset{\circ}{P_3}(\bar{z})\right\}\right) < \kappa_3. \tag{2}$$

This implies the claim for $I = \{1\}$, $I = \{3\}$ and $I = \{1, 3\}$.

It remains to consider the cases $I = \{0, 2\}$ and $0 \in I \subseteq \{0, 1, 3\}$. We need some preliminary considerations. Let

$$\varepsilon := \tfrac{1}{2} \min\{d((x(c), y(c)), P_i(\bar{z})) : c \in C,\ i \in \{0, 1, 2, 3\},\ (x(c), y(c)) \notin P_i(\bar{z})\}$$

and $\tilde{z} := \bar{z} - \varepsilon$. Observe that $w_1(\tilde{z}) \in \{w_1(\bar{z}) - \varepsilon, w_1(\bar{z}) + \varepsilon\}$ and $w_2(\tilde{z}) \in \{w_2(\bar{z}) - \varepsilon, w_2(\bar{z}) + \varepsilon\}$ (see Figs. 7–9). The minimality of $\bar{z}$ implies that

$$\text{size}\left(\left\{c \in C : (x(c), y(c)) \in \bigcup_{j \in J} P_j(\tilde{z})\right\}\right) < \sum_{j \in J} \kappa_j \tag{*}$$

holds for at least one $0 \in J \subseteq \{0, 1, 3\}$.

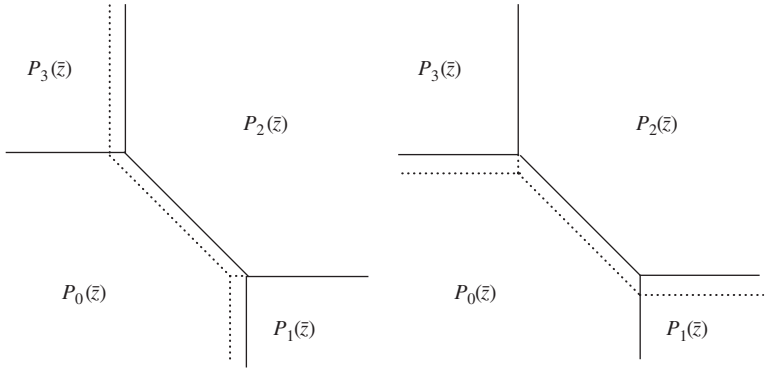

Fig. 8.

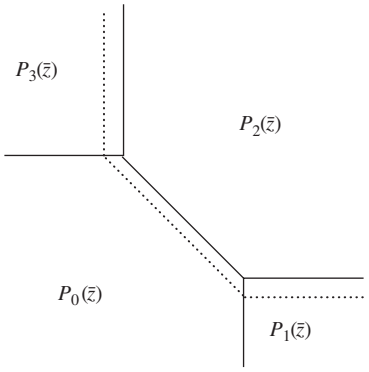

Fig. 9.

This is possible only if

$$\text{size}(\{c \in C : (x(c), y(c)) \in \overset{\circ}{P_0}(\bar{z})\}) < \kappa_0. \tag{3}$$

This directly implies the claim for $I = \{0\}$.

Now, the claim for $I = \{0, 2\}$ is equivalent to

$$\text{size}(\{c \in C : (x(c), y(c)) \in P_1(\bar{z}) \cup P_3(\bar{z})\}) \geqslant \kappa_1 + \kappa_3.$$

This inequality can be violated only if $P_1(\bar{z}) \cap P_3(\bar{z}) \neq \emptyset$, i.e. $w_1(\bar{z}) = w_2(\bar{z})$. But then we have $P_0(\bar{z}) \cup P_1(\bar{z}) \cup P_3(\bar{z}) = \overset{\circ}{P_0}(\bar{z}) \,\dot\cup\, (P_1(\bar{z}) \cup P_3(\bar{z}))$ and thus

$$\kappa_0 + \kappa_1 + \kappa_3 \leqslant \text{size}(\{c \in C : (x(c), y(c)) \in \overset{\circ}{P_0}(\bar{z}) \,\dot\cup\, (P_1(\bar{z}) \cup P_3(\bar{z}))\}),$$

which, together with (3), implies the claim for $I = \{0, 2\}$.

Finally, to prove the claim for the cases $I = \{0, 1\}$, $I = \{0, 3\}$ and $I = \{0, 1, 3\}$, we distinguish four cases.

Case 1: $w_1(\bar{z}) = w_1(\bar{z}) + \varepsilon$ and $w_2(\bar{z}) = w_2(\bar{z}) - \varepsilon$. Then

$$\text{size}(\{c \in C : (x(c), y(c)) \in P_1(\bar{z}) \setminus P_2(\bar{z})\}) < \kappa_1 \tag{4}$$

and

$$\text{size}(\{c \in C : (x(c), y(c)) \in P_3(\bar{z}) \setminus P_2(\bar{z})\}) < \kappa_3 \tag{5}$$

(see Fig. 7; dotted lines correspond to $\tilde{z}$). The claim for $I = \{0, 1\}$ follows from (3) and (4), for $I = \{0, 3\}$ from (3) and (5), for $I = \{0, 1, 3\}$ from (3) to (5).

Case 2: $w_1(\tilde{z}) = w_1(\bar{z}) + \varepsilon$ and $w_2(\tilde{z}) = w_2(\bar{z}) + \varepsilon$. Then

$$\text{size}(\{c \in C : (x(c), y(c)) \in P_1(\bar{z}) \setminus P_2(\bar{z})\}) < \kappa_1 \tag{4}$$

and

$$\text{size}(\{c \in C : (x(c), y(c)) \in P_3(\bar{z}) \setminus P_2(\bar{z})\}) \geqslant \kappa_3 \tag{$\neg$5}$$

(see Fig. 8, left-hand side). Because of $(*)$ at least one of the following two statements holds:

$$\text{size}(\{c \in C : (x(c), y(c)) \in P_0(\bar{z}) \setminus (P_1(\bar{z}) \cup P_2(\bar{z}))\}) < \kappa_0, \tag{3$'$}$$

$$\text{size}(\{c \in C : (x(c), y(c)) \in (P_0(\bar{z}) \cup P_3(\bar{z})) \setminus (P_1(\bar{z}) \cup P_2(\bar{z}))\}) < \kappa_0 + \kappa_3. \tag{6}$$

In fact, we have (6) in each case, as this is also implied by (3′) together with (2).

For $I = \{0, 1\}$ the claim now follows from (3) and (4), for $I = \{0, 3\}$ from (6), and for $I = \{0, 1, 3\}$ from (4) and (6).

Case 3: $w_1(\tilde{z}) = w_1(\bar{z}) - \varepsilon$ and $w_2(\tilde{z}) = w_2(\bar{z}) - \varepsilon$. This case is symmetric to Case 2 (see Fig. 8, right-hand side).

Case 4: $w_1(\tilde{z}) = w_1(\bar{z}) - \varepsilon$ and $w_2(\tilde{z}) = w_2(\bar{z}) + \varepsilon$. Then

$$\text{size}(\{c \in C : (x(c), y(c)) \in P_1(\bar{z}) \setminus P_2(\bar{z})\}) \geqslant \kappa_1 \tag{$\neg$4}$$

and

$$\text{size}(\{c \in C : (x(c), y(c)) \in P_3(\bar{z}) \setminus P_2(\bar{z})\}) \geqslant \kappa_3 \tag{$\neg$5}$$

(see Fig. 9). Moreover, because of $(*)$ at least one of the following four statements holds:

$$\text{size}(\{c \in C : (x(c), y(c)) \in P_0(\bar{z}) \setminus P_2(\bar{z})\}) < \kappa_0, \tag{3$''$}$$

$$\text{size}(\{c \in C : (x(c), y(c)) \in (P_0(\bar{z}) \cup P_3(\bar{z})) \setminus P_2(\bar{z})\}) < \kappa_0 + \kappa_3, \tag{6$'$}$$

$$\text{size}(\{c \in C : (x(c), y(c)) \in (P_0(\bar{z}) \cup P_1(\bar{z})) \setminus P_2(\bar{z})\}) < \kappa_0 + \kappa_1, \tag{7$'$}$$

$$\text{size}(\{c \in C : (x(c), y(c)) \in (P_0(\bar{z}) \cup P_1(\bar{z}) \cup P_3(\bar{z})) \setminus P_2(\bar{z})\}) < \kappa_0 + \kappa_1 + \kappa_3. \tag{8}$$

This implies the three inequalities:

$$\text{size}(\{c \in C : (x(c), y(c)) \in (P_0(\bar{z}) \cup P_3(\bar{z})) \setminus (P_1(\bar{z}) \cup P_2(\bar{z}))\}) < \kappa_0 + \kappa_3, \tag{6}$$

$$\text{size}(\{c \in C : (x(c), y(c)) \in (P_0(\bar{z}) \cup P_1(\bar{z})) \setminus (P_2(\bar{z}) \cup P_3(\bar{z}))\}) < \kappa_0 + \kappa_1 \tag{7}$$

and (8), as is easily seen: (3″) and (2) imply (6′), (6′) and (1) imply (8), and (7′) and (2) imply (8). Moreover, (8) and ($\neg$4) imply (6), and (8) and ($\neg$5) imply (7).

The three statements (6), (7) and (8) directly imply the claim for $I = \{0, 3\}$, $I = \{0, 1\}$ and $I = \{0, 1, 3\}$. □

We have reached the goal of this section.

Theorem 6.4. *There always exists a feasible American map.*

Proof. By Lemma 6.2 at least one of $(P_0(\bar{z}), P_1(\bar{z}), P_2(\bar{z}), P_3(\bar{z}))$ and $(Q_0(\bar{w}), Q_1(\bar{w}), Q_2(\bar{w}), Q_3(\bar{w}))$ is an American map. This map is feasible by Theorem 5.3, as the condition mentioned there is satisfied by Lemma 6.3. □

7. Optimum fractional partitions

In this section, we show that a feasible fractional partition that is consistent with an American map must be optimum. The following optimality criterion will be useful:

Lemma 7.1. *Let g, g' be feasible fractional partitions. Consider the complete directed graph on vertices $\{0,1,2,3\}$ and edge weights $s : E(G) \to \mathbb{R}$ defined by*

$$s((i,j)) := \max\{d((x(c),y(c)),R_j) - d((x(c),y(c)),R_i) : g'(c,j) > g(c,j),\ \ g'(c,i) < g(c,i)\}$$

for $i, j \in \{0,1,2,3\}$ with $i \neq j$ (where $\max \emptyset := -\infty$). If G has no directed circuit with positive total weight, then the total movement of g' is smaller than or equal to the total movement of g.

Proof. Assuming that no directed circuit of G has positive total weight, we have to show that

$$\Delta := \sum_{c \in C} \text{size}(c) \sum_{i=0}^{3} (g'(c,i) - g(c,i))\ d((x(c),y(c)),R_i) \leqslant 0.$$

We may assume that g and g' are integral; otherwise we split elements of C (i.e. replace an element $c \in C$ by two elements c^-, c^+ with $x(c^-) = x(c^+) = x(c)$, $y(c^-) = y(c^+) = y(c)$ and $\text{size}(c^-) + \text{size}(c^+) = \text{size}(c)$).

Consider the circulation f in G that is defined by

$$f((i,j)) := \text{size}(\{c \in C : g'(c,j) = 1,\ \ g(c,i) = 1\}).$$

f is indeed a circulation because both g and g' satisfy the capacity constraints. Obviously $\Delta \leqslant \sum_{e \in E(G)} s(e) f(e)$. But the right-hand side is the weight of the circulation, which cannot be positive as no directed circuit in G has positive total weight. □

Theorem 7.2. *Any feasible fractional partition of C that is consistent with some American map must be optimum.*

Proof. Let $\mathcal{L} = (L_0, L_1, L_2, L_3)$ an American map, and let g' a feasible fractional partition of C consistent with $\mathcal{L}$. W.l.o.g. let $\mathcal{L}$ be an American map of the first type (of Definition 5.4), i.e. there are numbers z, w_1, w_2 with $w_1 \leqslant w_2$ and $\mathcal{L} = \mathcal{L}(z, w_1, w_2)$.

Let now g be any optimum fractional partition. We use Lemma 7.1 to prove that the total movement of g' is not greater than that of g. For the edge weights s of the complete directed graph on $\{0,1,2,3\}$, as defined in Lemma 7.1, we have by definition

$$s((i,j)) \leqslant \max\{d((x,y),R_j) - d((x,y),R_i) : (x,y) \in L_j\}$$

for each edge (i,j).

This maximum is attained in those points $(x,y) \in L_j$, for which $d((x,y),L_i)$ is minimal.

Let

$$\alpha := \frac{z - w_1}{2} - x_1,$$
$$\beta := y_1 - \frac{z + w_1}{2},$$
$$\gamma := x_1 - \frac{z - w_2}{2},$$
$$\delta := \frac{z + w_2}{2} - y_1$$

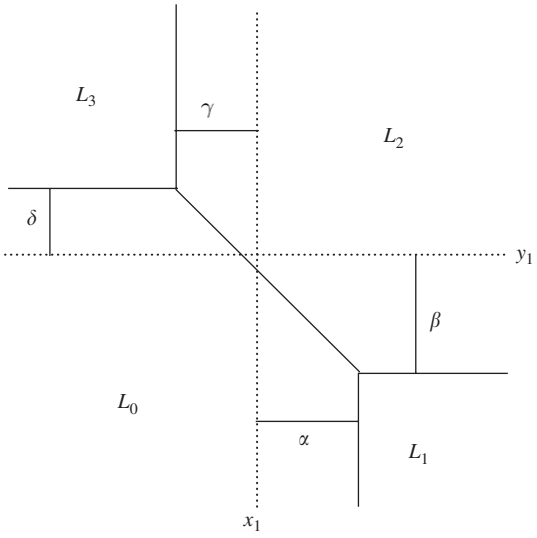

Fig. 10.

(cf. Fig. 10). Of course, some of the numbers α, β, γ and δ can be negative. As upper bounds on the edge weights we get

$$
\begin{aligned}
&s((1,0)) \leqslant \alpha, \quad s((0,1)) \leqslant -\alpha,\\
&s((1,2)) \leqslant \beta, \quad s((2,1)) \leqslant -\beta,\\
&s((3,2)) \leqslant \gamma, \quad s((2,3)) \leqslant -\gamma,\\
&s((3,0)) \leqslant \delta, \quad s((0,3)) \leqslant -\delta,\\
&s((2,0)) \leqslant \alpha - \beta, \quad s((0,2)) \leqslant \beta - \alpha,\\
&s((3,1)) \leqslant -\alpha - \beta, \quad s((1,3)) \leqslant -\gamma - \delta.
\end{aligned}
$$

The maximal edge weights are illustrated by Fig. 11. With the above weights there exists no directed circuit with positive total weight, because

$$\alpha + \gamma = \beta + \delta = \frac{w_2 - w_1}{2} \geqslant 0.$$

By Lemma 7.1 this implies that g' is also an optimum fractional partition. □

We conclude:

Theorem 7.3. *There always exists an optimum fractional partition of C which is consistent with some American map.*

Proof. This follows directly from Theorems 6.4 and 7.2. □

Figs. 14 and 15 (in the Appendix) show two real-world examples, where the American map structure can be seen nicely. The small objects to be distributed are colored according to their assignment in the optimum partition. The large (gray) objects have been fixed previously.

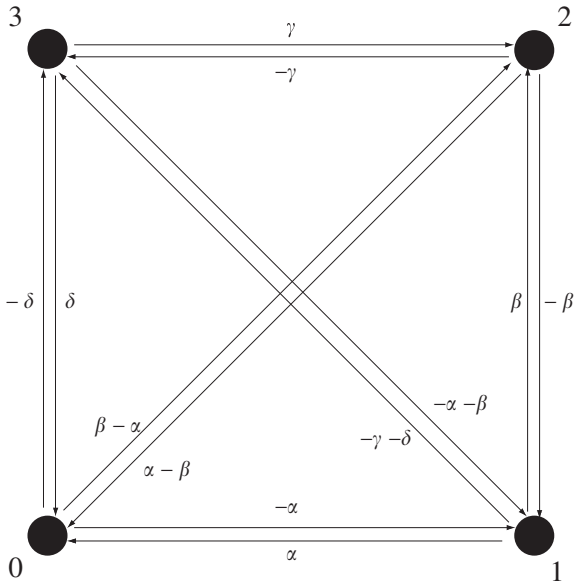

Fig. 11.

The linear-time algorithm of the next section will build on the above proof, in particular on the results of Section 6. But we should remark that Theorem 7.3 has a simpler proof using LP duality, which we sketch now:

Let g, λ, μ be an optimum pair of solutions to the primal-dual pair of LPs

$$\begin{aligned}\min \Bigg\{ &\sum_{c\in C}\sum_{i=0}^{3} \mathrm{size}(c)d((x(c),y(c)),R_i)g(c,i) : g \geqslant 0,\\ &\sum_{c\in C} g(c,i)\mathrm{size}(c) \leqslant \kappa_i \ (i=0,\dots,3),\ \sum_{i=0}^{3} g(c,i) = 1 (c\in C)\Bigg\}\\ = \max \Bigg\{ &\sum_{i=0}^{3} \kappa_i\lambda_i + \sum_{c\in C}\mu_c : \lambda_i \leqslant 0 \ (i=0,\dots,3),\\ &\mu_c + \mathrm{size}(c)\lambda_i \leqslant \mathrm{size}(c)d((x(c),y(c)),R_i)\ (c\in C, i=0,\dots,3)\Bigg\}.\end{aligned}$$

Then, by complementary slackness, g, μ is an optimum pair of solutions to the primal-dual pair of LPs

$$\begin{aligned}\min \Bigg\{ &\sum_{c\in C}\sum_{i=0}^{3} \mathrm{size}(c)(d((x(c),y(c)),R_i) - \lambda_i)g(c,i) : g \geqslant 0,\ \sum_{i=0}^{3} g(c,i) = 1\ (c\in C)\Bigg\}\\ = \max \Bigg\{ &\sum_{c\in C}\mu_c : \mu_c \leqslant \mathrm{size}(c)d((x(c),y(c)),R_i) - \mathrm{size}(c)\lambda_i\ (c\in C, i=0,\dots,3)\Bigg\}.\end{aligned}$$

By the simple nature of this primal LP one can show with a bit of case checking that g is consistent with an American map: if $\lambda_0 + \lambda_2 \geqslant \lambda_1 + \lambda_3$, then we have an American map of the first type, otherwise of the second type.

8. Linear-time algorithm

The special structure of American maps enables us to find an optimum fractional partition in linear time. We again assume w.l.o.g. $\kappa_0, \kappa_1, \kappa_2, \kappa_3 > 0$ and $\text{size}(C) = \sum_{i=0}^{3} \kappa_i$.

As a subroutine, we shall often use an algorithm for weighted median computation (see Section 4). For $C' \subseteq C$, $f : C \to \mathbb{R}$ and $0 \leqslant \kappa \leqslant \text{size}(C')$ we write

$$C'' := \text{median}\,(C', f, \kappa)$$

and mean that $C'' \subseteq C'$ with $\text{size}(C'') = \kappa$ and $f(c_1) \leqslant f(c_2)$ for all $c_1 \in C''$, $c_2 \in C' \setminus C''$.

To guarantee the existence of such a set C'' (and to find it in linear time) we allow an element $c \in C'$ to be split up into two, i.e. replacing C' by $(C' \setminus \{c\}) \dot\cup \{c^-, c^+\}$, where $x(c^-) = x(c^+) = x(c)$, $y(c^-) = y(c^+) = y(c)$ and $\text{size}(c^-) + \text{size}(c^+) = \text{size}(c)$.

Evidently, C'' then corresponds to a feasible fractional partition of C' with respect to capacities κ and $\text{size}(C') - \kappa$, where at most one element has a non-integral value. C'' (and, if necessary, the modified C') can be computed in $O(|C'|)$ time with the algorithm mentioned in Section 4.

When we take the union of certain sets we rejoin elements which have been split up before. By this we shall be able to guarantee that no element is ever split up into more than 11 parts.

In addition, we shall often compute a non-weighted median. We write

$$C'' := \text{median}\,(C', f)$$

and mean a subset C'' of C' with $|C''| = \lceil |C'|/2 \rceil$ and $f(c_1) \leqslant f(c_2)$ for all $c_1 \in C''$, $c_2 \in C' \setminus C''$. So here no element is split up.

Much of the idea behind the algorithm is contained in the proof of Theorem 6.4. Among the two basic types of American maps (see Section 6) we first assume the first one, i.e. $(P_0(\bar z), P_1(\bar z), P_2(\bar z), P_3(\bar z))$. $\bar z$ is determined by a kind of binary search, where for each intermediate $\tilde z$ considered we compute the numbers $w_1(\tilde z)$ and $w_2(\tilde z)$ as weighted medians. By appropriate choice of $\tilde z$ in the possible interval $[z_{\min}, z_{\max}]$ one can ensure that the sets on which the medians are computed become smaller by a constant factor every five iterations; this will lead to a linear running time.

When we have found $\bar z$, we test whether $w_1(\bar z) \leqslant w_2(\bar z)$. If not, we start the analogous algorithm for the second basic type, i.e. we find $(Q_0(\bar w), Q_1(\bar w), Q_2(\bar w), Q_3(\bar w))$. By Lemma 6.2, we then have $z_1(\bar w) \leqslant z_2(\bar w)$. Hence, we find a feasible American map in any case.

As will become clear later, the presentation of the algorithm simplifies a lot if we check in advance the existence of a feasible map of the form $\mathscr{L}(z, w, w)$ (this is easy). If we do so, we can abort the main algorithm when discovering that $w_1(\bar z) \geqslant w_2(\bar z)$, and proceed to the second case (because we then know that $z_1(\bar w) < z_2(\bar w)$).

In the formal description of the algorithm let $f_z(c) := \max\{y(c), z - x(c)\}$ and $g_z(c) := \max\{x(c), z - y(c)\}$ for $c \in C$ and $z \in \mathbb{R}$.

① Set $i := 0$ and

$$\begin{aligned}
A &:= \text{median}\,(C, x + y, \kappa_0),\\
B &:= \text{median}\,(C \setminus A, -x - y, \kappa_2),\\
z_{\min} &:= \max\{(x + y)(c) : c \in A\},\\
z_{\max} &:= \min\{(x + y)(c) : c \in B\},\\
DE &:= \text{median}\,(C, f_{z_{\min}}, \kappa_1),\\
EF &:= \text{median}\,(C, f_{z_{\max}}, \kappa_1),
\end{aligned}$$

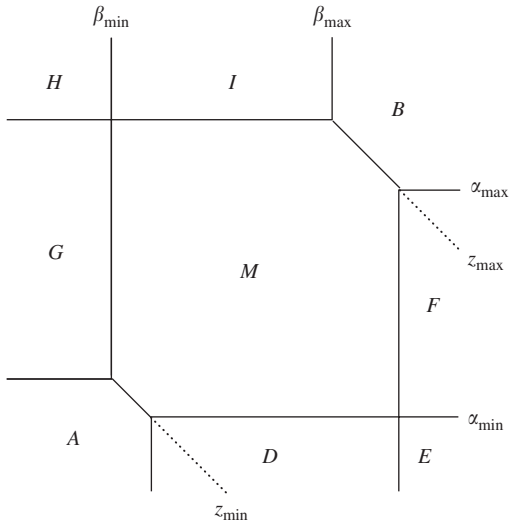

Fig. 12.

$GH := \text{median}\,(C, g_{z_{\min}}, \kappa_3)$,
$HI := \text{median}\,(C, g_{z_{\max}}, \kappa_3)$,
$\alpha_{\min} := \max\{f_{z_{\min}}(c) : c \in DE\}$,
$\alpha_{\max} := \max\{f_{z_{\max}}(c) : c \in EF\}$,
$\beta_{\min} := \max\{g_{z_{\min}}(c) : c \in GH\}$,
$\beta_{\max} := \max\{g_{z_{\max}}(c) : c \in HI\}$,
$A := A \setminus (DE \cup GH)$,
$B := B \setminus (EF \cup HI)$,
$D := DE \setminus EF$,
$E := DE \cap EF$,
$F := EF \setminus DE$,
$G := GH \setminus HI$,
$H := GH \cap HI$,
$I := HI \setminus GH$,
$M := C \setminus (A \cup B \cup D \cup E \cup F \cup G \cup H \cup I)$.

(see Fig. 12, but note that the sets D and G, and similarly F and I, need not be disjoint. The sets A, B, E and H contain those elements, which we can already be sure to be in $P_0(\bar{z})$, $P_2(\bar{z})$, $P_1(\bar{z})$ or $P_3(\bar{z})$, respectively).

② If $\alpha_{\min} \geqslant z_{\max} - \beta_{\max}$ or $\beta_{\min} \geqslant z_{\max} - \alpha_{\max}$, then stop (we have $w_1(\bar{z}) \geqslant w_2(\bar{z})$).

③ If $|M| \leqslant 1$ and $D = F = G = I = \emptyset$:

Set $g(c, 1) := 1$ for $c \in E$, $g(c, 3) := 1$ for $c \in H$, $g(c, 0) := 1$ for $c \in A$,
$g(c, 2) := 1$ for $c \in B$, $g(c, 0) := 1 - g(c, 2) := (\kappa_0 - \text{size}(A))/\text{size}(c)$ for $c \in M$, and stop.

④ If $z_{\min} = z_{\max}$:
Set $z := z_{\min}$, $w_1 := 2\alpha_{\min} - z_{\min}$, $w_2 := z_{\min} - 2\beta_{\min}$, $(L_0, L_1, L_2, L_3) := \mathscr{L}(z, w_1, w_2)$, determine g according to Theorem 5.3 and stop.

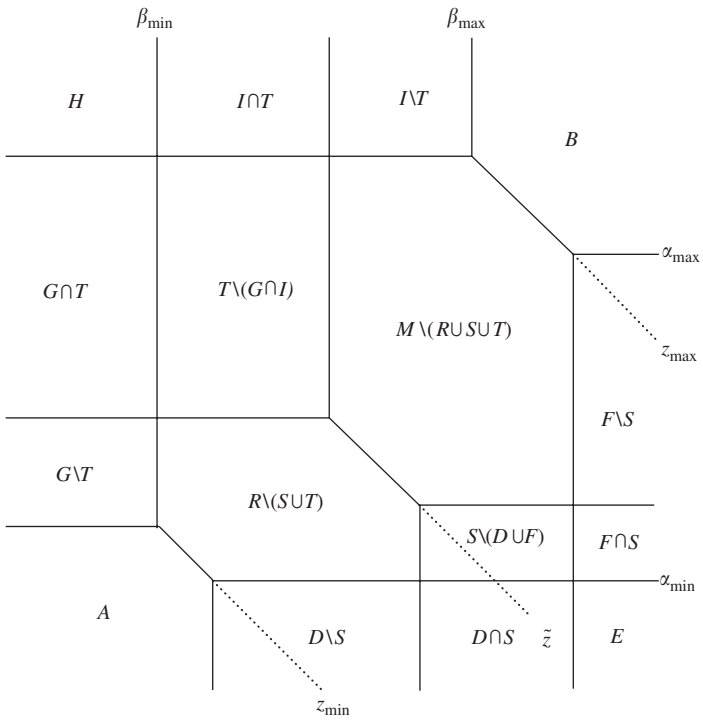

Fig. 13.

⑤ If $i \bmod 5 = 0$: If $M = \emptyset$, then go to ⑧. Otherwise, set

$$R := \text{median}\,(M, x + y),$$
$$\tilde{z} := \max\{(x + y)(c) : c \in R\}$$

and compute S and T (cf. Fig. 13 and the subroutines below).
If $i \bmod 5 = 1$: If $D = \emptyset$, then go to ⑧. Otherwise, set

$$U := \text{median}\,(D, -x),$$
$$\tilde{x} := \min\{x(c) : c \in U\},$$
$$V := \{c \in (G \cup I \cup M)\setminus U : x(c) > \tilde{x}\},$$
$$S := U \cup \text{median}\,(V \cup F, y, \kappa_1 - \text{size}(E) - \text{size}(U)),$$
$$\tilde{z} := \tilde{x} + \min\{y(c) : c \in (V \cup F)\setminus S\},$$
$$R := \{c \in M : (x + y)(c) \leqslant \tilde{z}\}$$

and compute T.
If $i \bmod 5 = 2$: If $F = \emptyset$, then go to ⑧. Otherwise, set

$$U := \text{median}\,(F, y),$$
$$\tilde{y} := \max\{y(c) : c \in U\},$$
$$V := \{c \in (G \cup I \cup M)\setminus U : y(c) < \tilde{y}\},$$
$$S := U \cup \text{median}\,(V \cup D, -x, \kappa_1 - \text{size}(E) - \text{size}(U)),$$
$$\tilde{z} := \tilde{y} + \max\{x(c) : c \in (V \cup D)\setminus S\},$$
$$R := \{c \in M : (x + y)(c) \leqslant \tilde{z}\}$$

and compute T.

If $i \bmod 5 = 3$: If $G = \emptyset$, then go to ⑧. Otherwise, set

$$\begin{aligned}
&U := \text{median}\,(G, -y),\\
&\tilde{y} := \min\{y(c) : c \in U\},\\
&V := \{c \in (D \cup F \cup M)\backslash U : y(c) > \tilde{y}\},\\
&T := U \cup \text{median}\,(V \cup I, x, \kappa_3 - \text{size}(H) - \text{size}(U)),\\
&\tilde{z} := \tilde{y} + \min\{x(c) : c \in (V \cup I)\backslash T\},\\
&R := \{c \in M : (x + y)(c) \leqslant \tilde{z}\}
\end{aligned}$$

and compute S.
If $i \bmod 5 = 4$: If $I = \emptyset$, then go to ⑧. Otherwise, set

$$\begin{aligned}
&U := \text{median}\,(I, x),\\
&\tilde{x} := \max\{x(c) : c \in U\},\\
&V := \{c \in (D \cup F \cup M)\backslash U : x(c) < \tilde{x}\},\\
&T := U \cup \text{median}\,(V \cup G, -y, \kappa_3 - \text{size}(H) - \text{size}(U)),\\
&\tilde{z} := \tilde{x} + \max\{y(c) : c \in (V \cup G)\backslash T\},\\
&R := \{c \in M : (x + y)(c) \leqslant \tilde{z}\}
\end{aligned}$$

and compute S.

The subroutines "Compute S" and "Compute T" are realized as follows:

Procedure "*Compute S*":
If $\alpha_{\min} > \tilde{z} - z_{\max} + \alpha_{\max}$, then set $W := \{c \in D : x(c) > \tilde{z} - \alpha_{\min}\}$, else set $W := \{c \in F : y(c) < \tilde{z} - z_{\max} + \alpha_{\max}\}$.
Set $S := W \cup \text{median}\,((D \cup F \cup G \cup I \cup M)\backslash W, f_{\tilde{z}}, \kappa_1 - \text{size}(E) - \text{size}(W))$.
Procedure "*Compute T*":
If $\beta_{\min} > \tilde{z} - z_{\max} + \beta_{\max}$, then set $W := \{c \in G : y(c) > \tilde{z} - \beta_{\min}\}$, else set $W := \{c \in I : x(c) < \tilde{z} - z_{\max} + \beta_{\max}\}$.
Set $T := W \cup \text{median}\,((D \cup F \cup G \cup I \cup M)\backslash W, g_{\tilde{z}}, \kappa_3 - \text{size}(H) - \text{size}(W))$.

⑥ Set

$$\begin{aligned}
&\tilde{\alpha} := \max\left\{\alpha_{\min}, \alpha_{\max} + \tilde{z} - z_{\max}, \max\{f_{\tilde{z}}(c) : c \in S\}\right\},\\
&\tilde{\beta} := \max\left\{\beta_{\min}, \beta_{\max} + \tilde{z} - z_{\max}, \max\{g_{\tilde{z}}(c) : c \in T\}\right\}.
\end{aligned}$$

⑦ If $\text{size}(A) + \text{size}((R \cup D \cup G)\backslash(S \cup T)) < \kappa_0$, then set

$$\begin{aligned}
&z_{\min} := \tilde{z},\\
&\alpha_{\min} := \tilde{\alpha},\\
&\beta_{\min} := \tilde{\beta},\\
&A := A \cup (R \cup D \cup G)\backslash(S \cup T),\\
&E := E \cup (S \cap F),\\
&D := S\backslash E,\\
&F := F\backslash E,\\
&H := H \cup (T \cap I),\\
&G := T\backslash H,\\
&I := I\backslash H,\\
&M := M\backslash(R \cup S \cup T).
\end{aligned}$$

If $\text{size}(A) + \text{size}((R \cup D \cup G)\setminus(S \cup T)) \geqslant \kappa_0$, then set

$$
\begin{aligned}
z_{\max} &:= \tilde{z},\\
\alpha_{\max} &:= \tilde{\alpha},\\
\beta_{\max} &:= \tilde{\beta},\\
B &:= B \cup ((M\setminus R) \cup F \cup I)\setminus(S \cup T),\\
E &:= E \cup (S \cap D),\\
D &:= D\setminus E,\\
F &:= S\setminus E,\\
H &:= H \cup (T \cap G),\\
G &:= G\setminus H,\\
I &:= T\setminus H,\\
M &:= R\setminus(S \cup T).
\end{aligned}
$$

⑧ Set $i := i + 1$ and go to ②.

This concludes the formal description of the algorithm. While ① serves as initialization and ②–④ contain the stopping criteria (see below), ⑤ and ⑦make up the core of the algorithm. Therefore, we first examine these a bit more.

Step ⑤ divides into five cases, four of which are symmetric to each other. It is the goal to halve the size of one of the sets M, D, F, G, I, for eventually (almost) all elements shall be assigned to one of the sets A, B, E, H.

If in ⑤ we have $i \bmod 5 = 0$, then the diagonal line $\tilde{z}$ is chosen such that (approximately) half of the elements of M are to the lower left of this line. Then (in the procedures "Compute S" and "Compute T") the numbers $w_1(\tilde{z})$ and $w_2(\tilde{z})$ are determined. Next, in ⑦ it will be decided whether $\tilde{z}$ replaces $z_{\max}$ as the new upper bound or $z_{\min}$ as the new lower bound. This decision will guarantee $z_{\min} \leqslant \tilde{z} \leqslant z_{\max}$, at least if $w_1(\bar{z}) < w_2(\bar{z})$ (see Lemma 8.5). Finally, the sets A, B, D, E, F, G, H, I, M are updated in ⑦; the numbers $\alpha_{\min}$, $\alpha_{\max}$, $\beta_{\min}$, $\beta_{\max}$ are needed only for the stopping criteria.

The case $i \bmod 5 = 1$ (and the other three, symmetric, cases) is a bit more complicated. Here, we first determine a coordinate $\tilde{x}$ such that approximately half of the elements of D are to the right. These will belong to S in any case. Then we add to S other objects to the right of $\tilde{x}$ (these can be elements of F and M, but also of G and I) until we have $\text{size}(S) + \text{size}(E) = \kappa_1$. Of course, this is done from bottom to top, i.e. we add to S all elements below a certain coordinate $\tilde{y}$. This then determines $\tilde{z} := \tilde{x} + \tilde{y}$ and $w_1(\tilde{z})$. As above, we finally use the procedure "Compute T" to determine $w_2(\tilde{z})$.

We add two more remarks: while in the case $i \bmod 5 = 1$ of ⑤ $\tilde{x}$ is the coordinate of the leftmost element of S, $\tilde{y}$ is the coordinate of the bottommost object that—though to the right of $\tilde{x}$—does not belong to S. This detail will help to guarantee that the algorithm terminates. Furthermore, observe that all operations (e.g. median searches) are performed only on the sets M, D, F, G, I. As these sets are reduced in each iteration an overall linear running time is possible.

Before analyzing the algorithm we remark the following fact, which follows directly from the definitions at the beginning of Section 6.

Lemma 8.1. *For arbitrary $z, z' \in \mathbb{R}$ we have $|w_1(z) - w_1(z')| \leqslant |z - z'|$ and $|w_2(z) - w_2(z')| \leqslant |z - z'|$.*

Now, we collect some invariants of the algorithm.

Lemma 8.2. *At any stage of the algorithm we have* $\text{size}(D) = \text{size}(F) = \kappa_1 - \text{size}(E)$ *and* $\text{size}(G) = \text{size}(I) = \kappa_3 - \text{size}(H)$. *Each time after ⑤ we also have* $\text{size}(S) = \kappa_1 - \text{size}(E)$ *and* $\text{size}(T) = \kappa_3 - \text{size}(H)$.

Lemma 8.3. *In each iteration of the algorithm we have $z_{\min} \leqslant \tilde{z} \leqslant z_{\max}$.*

Proof. In the case $i \bmod 5 = 0$ we obviously have $z_{\min} \leqslant \tilde{z} \leqslant z_{\max}$. In the case $i \bmod 5 = 1$ this follows from the fact that $F\setminus S \neq \emptyset$, which, in turn, is implied by $\text{size}(F\setminus S) = \kappa_1 - \text{size}(E) - \text{size}(F \cap S) = \text{size}(S\setminus F) \geqslant \text{size}(U)$ (cf. Lemma 8.2). The other cases are symmetric. □

Lemma 8.4. *In each iteration of the algorithm (after ⑥) we have*:

(a) $\tilde{\alpha} = \max\{f_{\tilde{z}}(c) : c \in S \cup E\}$ *and* $\tilde{\beta} = \max\{g_{\tilde{z}}(c) : c \in T \cup H\}$,
(b) $w_1(\tilde{z}) = 2\tilde{\alpha} - \tilde{z}$ *and* $w_2(\tilde{z}) = \tilde{z} - 2\tilde{\beta}$,
(c) $w_1(z_{\min}) = 2\alpha_{\min} - z_{\min}$ *and* $w_2(z_{\min}) = z_{\min} - 2\beta_{\min}$,
$w_1(z_{\max}) = 2\alpha_{\max} - z_{\max}$ *and* $w_2(z_{\max}) = z_{\max} - 2\beta_{\max}$,

unless the algorithm terminates immediately after that in ②, and we then have $(z_{\min}+w_1(z_{\min}))/2 \geqslant (z_{\max}+w_2(z_{\max}))/2$ *or* $(z_{\min} - w_2(z_{\min}))/2 \geqslant (z_{\max} - w_1(z_{\max}))/2$.

Proof. First, observe that (c) holds initially (after ①) by definition. To prove the claim by induction, we assume that (c) holds in a certain iteration of the algorithm (in ⑥). One easily checks that $\tilde{\alpha} \geqslant \max\{f_{\tilde{z}}(c) : c \in S \cup E\}$ and $\tilde{\beta} \geqslant \max\{g_{\tilde{z}}(c) : c \in T \cup H\}$.

Case 1: $\max\{z_{\max} - \beta_{\max}, \tilde{z} - \beta_{\min}\} \leqslant \max\{f_{\tilde{z}}(c) : c \in S \cup E\}$.

Then the algorithm terminates immediately in ②. Moreover, $(\tilde{z} + w_1(\tilde{z}))/2 \geqslant z_{\max} - \beta_{\max}$ and $(\tilde{z} - w_1(\tilde{z}))/2 \leqslant \beta_{\min}$. If ⑦ sets $z_{\min} := \tilde{z}$, then we still have $w_2(z_{\max}) = z_{\max} - 2\beta_{\max}$, implying $(z_{\min} + w_1(z_{\min}))/2 \geqslant (z_{\max} + w_2(z_{\max}))/2$. If ⑦ sets $z_{\max} := \tilde{z}$, then we still have $w_2(z_{\min}) = z_{\min} - 2\beta_{\min}$, implying $(z_{\max} - w_1(z_{\max}))/2 \leqslant (z_{\min} - w_2(z_{\min}))/2$.

Case 2: $\max\{z_{\max} - \alpha_{\max}, \tilde{z} - \alpha_{\min}\} \leqslant \max\{g_{\tilde{z}}(c) : c \in T \cup H\}$.

Again, the algorithm terminates immediately in ②. Analogously to Case 1 one proves that $(z_{\min} + w_1(z_{\min}))/2 \geqslant (z_{\max} + w_2(z_{\max}))/2$ or $(z_{\min} - w_2(z_{\min}))/2 \geqslant (z_{\max} - w_1(z_{\max}))/2$.

Case 3: Now, we show that otherwise (a), (b) and (after ⑦) (c) hold. We first prove

$$w_1(\tilde{z}) = 2\max\{f_{\tilde{z}}(c) : c \in S \cup E\} - \tilde{z}. \tag{*}$$

We start by noting that the choice of S in ⑤ of the algorithm ensures $\text{size}(S) = \kappa_1 - \text{size}(E)$ in all cases. Let $w' := 2\max\{f_{\tilde{z}}(c) : c \in S \cup E\} - \tilde{z}$. Since for all $c \in S \cup E$ we have

$$x(c) \geqslant \tilde{z} - f_{\tilde{z}}(c) \geqslant \tilde{z} - \frac{\tilde{z} + w'}{2} = \frac{\tilde{z} - w'}{2}$$

and

$$y(c) \leqslant f_{\tilde{z}}(c) \leqslant \frac{\tilde{z} + w'}{2},$$

we conclude that $w_1(\tilde{z}) \leqslant w'$. Moreover, for all $c' \in C$ with $f_{\tilde{z}}(c') < \max\{f_{\tilde{z}}(c) : c \in S \cup E\}$ we have $c' \in S \cup E$—otherwise we are in Case 1 or get a contradiction to Lemma 8.3. Hence,

$$\text{size}\left\{c \in C : x(c) \geqslant \frac{\tilde{z} - w}{2},\ y(c) \leqslant \frac{\tilde{z} + w}{2}\right\} < \text{size}(S \cup E) = \kappa_1$$

for all $w < w'$, implying $(*)$.

Analogously to $(*)$ we also have

$$w_2(\tilde{z}) = \tilde{z} - 2\max\{g_{\tilde{z}}(c) : c \in T \cup H\}.$$

Now, it suffices to prove (a), as this implies (b) by the above consideration, and then (c) also continues to hold after ⑦.

The choice of $\tilde{\alpha}$ implies $\tilde{\alpha} \geqslant \max\{f_{\tilde{z}}(c) : c \in S \cup E\}$. If the maximum in the definition of $\tilde{\alpha}$ is attained for some $c \in S$, we even have equality. Otherwise, we have $S \subseteq D$ or $S \subseteq F$, which (with Lemma 8.2) implies $S = D$ or $S = F$. In this case the induction hypothesis yields $\tilde{\alpha} = \max\{f_{\tilde{z}}(c) : c \in S \cup E\}$. The equation $\tilde{\beta} = \max\{g_{\tilde{z}}(c) : c \in T \cup H\}$ follows analogously. □

Lemma 8.5. *At any stage of the algorithm we have*:

(a) $\bar{z} \geqslant z_{\min}$ *or* $(w_1(z_{\min}) \geqslant w_2(z_{\min})$ *and* $w_1(\bar{z}) \geqslant w_2(\bar{z}))$;
(b) $\bar{z} \leqslant z_{\max}$ *or* $(w_1(z_{\max}) \geqslant w_2(z_{\max})$ *and* $w_1(\bar{z}) \geqslant w_2(\bar{z}))$.

Proof. (a) Initially (after ①) we have size($\{c \in C : x(c) + y(c) < z_{\min}\}$) $< \kappa_0$, implying $\bar{z} \geqslant z_{\min}$ directly.

If ⑦ increases $z_{\min}$ to $\tilde{z}$, then we have size($A \cup ((R \cup D \cup G) \setminus (S \cup T))$) $< \kappa_0$. This implies $\bar{z} \geqslant \tilde{z} - \max\{0, (w_1(\tilde{z}) - w_2(\tilde{z}))/2\}$. Together with Lemma 8.1 we then obtain (a).

(b) Initially (after ①) we have size($\{c \in C : x(c) + y(c) > z_{\max}\}$) $< \kappa_2$. In the case $w_1(z_{\max}) < w_2(z_{\max})$ this directly implies $\bar{z} \leqslant z_{\max}$, using size($A \cup D \cup G \cup M \cup \{c \in B : x(c) + y(c) = z_{\max}\}$) $> \kappa_0$. In the case $w_1(\bar{z}) < w_2(\bar{z})$ Lemma 6.3 implies size($\{c \in C : (x(c), y(c)) \in P_2(\bar{z})\}$) $\geqslant \kappa_2$, which also yields $\bar{z} \leqslant z_{\max}$. Hence, initially (b) holds.

If ⑦ sets $z_{\max} := \tilde{z}$, then we have size($\{c \in C : (x(c), y(c)) \in P_0(\tilde{z})\}$) $\geqslant$ size(A) + size($(R \cup D \cup G) \setminus (S \cup T)$) $\geqslant \kappa_0$, size($\{c \in C : (x(c), y(c)) \in P_1(\tilde{z})\}$) $\geqslant$ size($S \cup E$) $= \kappa_1$ and size($\{c \in C : (x(c), y(c)) \in P_3(\tilde{z})\}$) $\geqslant$ size($T \cup H$) $= \kappa_3$. If S and T are disjoint, then the sets $A \cup ((R \cup D \cup G) \setminus (S \cup T))$ and $S \cup E$ and $T \cup H$ are also pairwise disjoint, which—by definition of $\bar{z}$—implies that $\bar{z} \leqslant \tilde{z}$.

If S and T are not disjoint, then we have $w_1(\tilde{z}) \geqslant w_2(\tilde{z})$. Suppose then $\bar{z} > \tilde{z}$ and $w_1(\bar{z}) < w_2(\bar{z})$. By Lemma 8.1, we get $\bar{z} > \tilde{z} + (w_1(\tilde{z}) - w_2(\tilde{z}))/2$. But then $z' := \bar{z} - (w_2(\bar{z}) - w_1(\bar{z}))/2$ yields a contradiction to the minimality of $\bar{z}$. □

Lemma 8.6. *If the algorithm terminates in* ②, *we indeed have*

$$w_1(\bar{z}) \geqslant w_2(\bar{z}).$$

Proof. Due to Lemma 8.5 we only have to consider the case $z_{\min} \leqslant \bar{z} \leqslant z_{\max}$. We have $(z_{\min} + w_1(z_{\min}))/2 \geqslant (z_{\max} + w_2(z_{\max}))/2$ or $(z_{\min} - w_2(z_{\min}))/2 \geqslant (z_{\max} - w_1(z_{\max}))/2$: this follows from Lemma 8.4 either directly or from part (c) together with the stopping criterion.

From $z_{\min} \leqslant \bar{z} \leqslant z_{\max}$ and Lemma 8.1 we get in one case

$$\frac{\bar{z} + w_1(\bar{z})}{2} \geqslant \frac{z_{\min} + w_1(z_{\min})}{2} \geqslant \frac{z_{\max} + w_2(z_{\max})}{2} \geqslant \frac{\bar{z} + w_2(\bar{z})}{2},$$

in the other case

$$\frac{\bar{z} - w_2(\bar{z})}{2} \geqslant \frac{z_{\min} - w_2(z_{\min})}{2} \geqslant \frac{z_{\max} - w_1(z_{\max})}{2} \geqslant \frac{\bar{z} - w_1(\bar{z})}{2}.$$

This implies the assertion. □

Lemma 8.7. *If the algorithm terminates with a fractional partition g, then g is feasible and consistent with an American map.*

Proof. If the algorithm terminates in ③, then g is feasible and, because of $D = F = G = I = \emptyset$, consistent with an American map.

If the algorithm terminates in ④, then we conclude from Lemma 8.4(c) and the fact that immediately before in ② the algorithm did not terminate:

$$w_1(z_{\min}) = 2\alpha_{\min} - z_{\min} < 2z_{\max} - 2\beta_{\max} - z_{\min} = z_{\max} - 2\beta_{\max} = w_2(z_{\max}),$$

i.e. $w_1(z_{\min}) < w_2(z_{\min})$ and $w_1(z_{\max}) < w_2(z_{\max})$. So Lemma 8.5 yields $\bar{z} = z_{\min} = z_{\max}$, i.e. we have $(L_0, L_1, L_2, L_3) = (P_0(\bar{z}), P_1(\bar{z}), P_2(\bar{z}), P_3(\bar{z}))$. Applying Lemma 6.3 and Theorem 5.3 concludes the proof. □

By now, it is not clear that the algorithm terminates at all. However, this is the case after $O(\log |C|)$ iterations, and the overall running time is linear:

Lemma 8.8. *The algorithm terminates after* $O(|C|)$ *elementary computation steps.*

Proof. Let M_i, D_i, F_i, G_i, I_i be the sets M, D, F, G, I at the beginning of iteration i (say before executing ⑤). Evidently, $M_{i+1} \subseteq M_i$ and $D_{i+1} \cup F_{i+1} \cup G_{i+1} \cup I_{i+1} \subseteq D_i \cup F_i \cup G_i \cup I_i \cup M_i$ for all i. Furthermore, $D_i \subseteq D_{i+1} \cup A_{i+1} \cup E_{i+1}$, $F_i \subseteq F_{i+1} \cup B_{i+1} \cup E_{i+1}$, $G_i \subseteq G_{i+1} \cup A_{i+1} \cup H_{i+1}$ and $I_i \subseteq I_{i+1} \cup B_{i+1} \cup H_{i+1}$ for all i.

Let now $k \geqslant 0$, $k \bmod 5 = 0$. Then

$$|M_{k+5}| \leqslant |M_{k+1}| \leqslant \left\lceil \frac{|M_k|}{2} \right\rceil$$

and

$$|D_{k+5} \cup F_{k+5} \cup G_{k+5} \cup I_{k+5} \cup M_{k+5}| \leqslant |D_k \cup F_k \cup G_k \cup I_k \cup M_k| - \left\lfloor \frac{|D_k|}{2} \right\rfloor - \left\lfloor \frac{|F_k|}{2} \right\rfloor - \left\lfloor \frac{|G_k|}{2} \right\rfloor - \left\lfloor \frac{|I_k|}{2} \right\rfloor.$$

Let n_1 be the first iteration with $n_1 \bmod 5 = 0$, at the beginning of which $|D \cup F \cup G \cup I \cup M| \leqslant 9$. (If this is never the case, let n_1 be the last iteration with $n_1 \bmod 5 = 0$.) Then we have by the above inequality for all $k \leqslant n_1 - 10$ with $k \bmod 5 = 0$:

$$|M_{k+10} \cup D_{k+10} \cup F_{k+10} \cup G_{k+10} \cup I_{k+10}| \leqslant \tfrac{3}{4}(|M_k \cup D_k \cup F_k \cup G_k \cup I_k| + 3).$$

As the number of steps in iteration i depends linearly on $|D_i \cup F_i \cup G_i \cup I_i \cup M_i|$, this implies a linear overall running time up to iteration n_1. The possible splitting of elements in the weighted median search is irrelevant for the running time because no element is ever split up into more than eleven parts.

We now show that the algorithm terminates in iteration $n_1 + 55$ at the latest. Suppose that this is not true.

Obviously, in the beginning of iteration $n_1 + 25$ none of the sets M, D, F, G and I contains more than one element. Let now n_2 be minimal with $n_2 \bmod 5 = 0$, such that in iterations n_2 to $n_2 + 4$ none of the sets D, F, G, I is M modified. Evidently, $n_1 \leqslant n_2 \leqslant n_1 + 50$.

Suppose that $D_{n_2} \neq \emptyset$, say $D_{n_2} = \{c_1\}$, and thus, because of $\text{size}(D_{n_2}) = \text{size}(F_{n_2}) = \kappa_1 - \text{size}(E_{n_2})$, also $F_{n_2} \neq \emptyset$, say $F_{n_2} = \{c_2\}$. Then $M = \{c_3\}$ with $x(c_3) > x(c_1)$ and $y(c_3) < y(c_2)$ is impossible, as otherwise at least one of the sets D, F would be modified in iteration n_2. Therefore, iteration $n_2 + 1$ sets $z_{\min} := x(c_1) + y(c_2)$. But then iteration $n_2 + 2$ sets $z_{\max} := x(c_1) + y(c_2)$, and the algorithm terminates. We conclude that $D_{n_2} = F_{n_2} = \emptyset$.

Analogously one shows that $G_{n_2} = I_{n_2} = \emptyset$, as otherwise the algorithm terminates after iteration $n_2 + 4$, at the latest. But as we also have $|M_{n_2}| \leqslant 1$, the algorithm must have terminated already in iteration n_2.

A possible application of Theorem 5.3 in ④ before termination needs only $O(|C|)$ time, hence the total running time is also $O(|C|)$. □

Theorem 8.9. *An optimum fractional partition can be found in linear time.*

Proof. As usual we assume $\kappa_0, \kappa_1, \kappa_2, \kappa_3 > 0$. We first check whether there is a feasible map of the form $\mathscr{L}(z, w, w)$. This can be done very easily by two weighted median computations. In the affirmative case we are done by Theorems 5.3 and 7.2.

Otherwise, we know that $w_1(\bar{z}) \neq w_2(\bar{z})$ and $z_1(\bar{w}) \neq z_2(\bar{w})$. We run the above algorithm, which has a linear running time by Lemma 8.8. If it terminates in ③ or ④ with a fractional partition g, then this is optimum by Lemma 8.7 and Theorem 7.2. Otherwise, we conclude from Lemma 8.6 that $w_1(\bar{z}) > w_2(\bar{z})$. We run the algorithm for the instance mirrored at the vertical axis. As by Lemma 6.2 we have $z_1(\bar{w}) < z_2(\bar{w})$, we now get an American map and a feasible fractional partition consistent with it. □

Instead of the sets M, D, F, G and I taking turns in being halved in ⑤, one could also operate on the one which has largest cardinality in each iteration. This also leads to a linear running time and may be a bit faster in practice.

The algorithm described in this paper has been implemented and proved to be very efficient in practice. It is a major component of several placement tools, used for the design of many industrial chips (e.g. see [22,6]). In particular, several processor series and most complex ASICs that have been designed by IBM recently (e.g. see [12,13]) have benefited directly from the work in this paper.

Acknowledgements

Many thanks to Christoph Albrecht for implementing the algorithm of Section 8, to Chuck Alpert, Ulrich Brenner, Jürgen Koehl, Bernhard Korte, and anonymous reviewers for their important comments, and to others at our institute and at IBM for the excellent cooperation.

9. Appendix. Real-world examples

Figs. 14 and 15 show examples from real designs. Large gray rectangles are replaced objects. Colored small objects are placed in order to minimize total squared interconnect length, regardless of overlaps. This so-called quadratic placement is the first step of many VLSI placement algorithms.

The chip area is divided into four regions of approximately equal size. The Quadrisection Problem is to assign the colored objects to the four regions with respect to their area capacities, such that the total movement is minimized.

The colors show the optimum solution: blue objects go to the upper left region, yellow ones to the upper right, red ones to the lower left, and green ones to the lower right region. The American map structure is clearly visible.

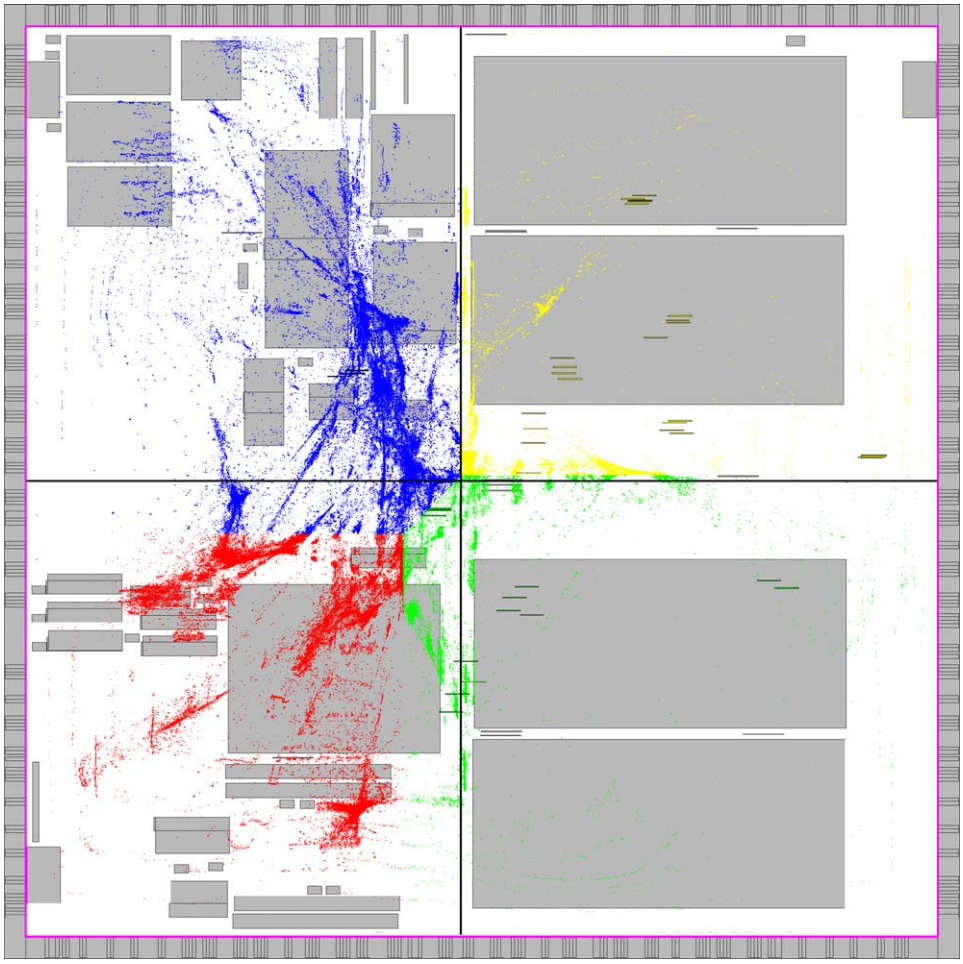

Fig. 14.

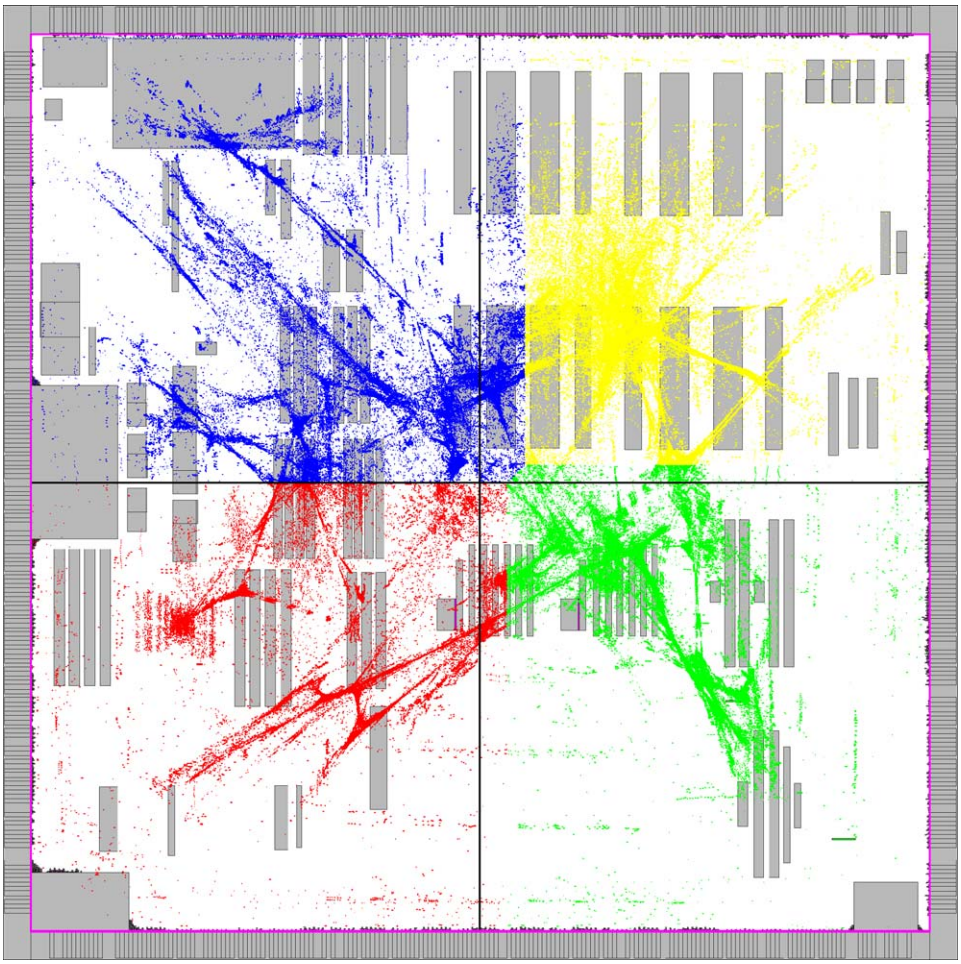

Fig. 15.

References

[1] R.K. Ahuja, J.B. Orlin, C. Stein, R.E. Tarjan, Improved algorithms for bipartite network flow, SIAM J. Comput. 23 (1994) 906–933.
[2] C.J. Alpert, A.B. Kahng, Recent directions in netlist partitioning: a survey, Integration, the VLSI J. 19 (1995) 1–81.
[3] M. Blum, R.W. Floyd, V. Pratt, R.L. Rivest, R.E. Tarjan, Time bounds for selection, J. Comput. System Sci. 7 (1973) 448–461.
[4] E. Boros, F.K. Hwang, Optimality of nested partitions and its application to cluster analysis, SIAM J. Optim. 6 (1996) 1153–1162.
[5] U. Brenner, Plazierung im VLSI-Design, Diploma Thesis, University of Bonn, 2000.
[6] U. Brenner, A. Rohe, An effective congestion driven placement framework, Proceedings of the ACM International Symposium on Physical Design, 2002, pp. 6–11.
[7] C. Chekuri, S. Khanna, A PTAS for the multiple knapsack problem, in: Proceedings of the Eleventh Annual ACM-SIAM Symposium on Discrete Algorithms, 2000, pp. 213–222.
[8] G.B. Dantzig, Discrete-variable extremum problems, Oper. Res. 5 (1957) 266–277.
[9] G. Even, J. Naor, S. Rao, B. Schieber, Divide-and-conquer approximation algorithms via spreading metrics, J. ACM 47 (2000) 585–616.
[10] K. Jansen, L. Porkolab, Improved approximation schemes for scheduling unrelated parallel machines, Math. Oper. Res. 26 (2001) 324–338.
[11] R.M. Karp, Reducibility among combinatorial problems, in: R.E. Miller, J.W. Thatcher (Eds.), Complexity of Computer Computations, Plenum Press, New York, 1972, pp. 85–103.
[12] J. Koehl, U. Baur, T. Ludwig, B. Kick, T. Pflueger, A flat, timing-driven design system for a high-performance CMOS processor chipset, in: Proceedings of the Conference on Design, Automation, and Test in Europe, IEEE Press, New York, 1998, pp. 312–320.
[13] J. Koehl, D.E. Lackey, G.W. Doerre, IBM's 50 million gate ASICs, in: Proceedings of the Asia and South Pacific Design Automation Conference (ASP-DAC), IEEE Press, New York, 2003, pp. 628–634.
[14] B. Korte, J. Vygen, Combinatorial Optimization: Theory and Algorithms, Springer, Berlin, 2000 (third ed. 2006).
[15] T. Lengauer, Combinatorial Algorithms for Integrated Circuit Layout, Teubner, Wiley, Chichester, 1990.

[16] J.K. Lenstra, D.B. Shmoys, É. Tardos, Approximation algorithms for scheduling unrelated parallel machines, Math. Programming 46 (1990) 259–271.
[17] J.B. Orlin, A faster strongly polynomial minimum cost flow algorithm, Oper. Res. 41 (1993) 338–350.
[18] D.B. Shmoys, É. Tardos, An approximation algorithm for the generalized assignment problem, Math. Programming 62 (1993) 461–474.
[19] T. Tokuyama, J. Nakano, Faster algorithms for the Hitchcock transportation problem, SIAM J. Comput. 24 (1995) 563–578.
[20] A.N. Tolstoĭ, Metody nakhozhdeniya naimen'shego summovogo kilometrazha pri planirovanii perevozok v prostanstve, in: Planirovanie Perevozok, Sbornik pervyĭ, Transpechat' NKPS, Moskow 1930, pp. 23–55 (see A. Schrijver, On the history of the transportation and maximum flow problems, Math. Programming, 91 (2002) 437–445).
[21] R.-S. Tsay, E.S. Kuh, C.-P. Hsu, Proud: a sea-of-gates placement algorithm, IEEE Design and Test of Comput. 5 (1988) 44–56.
[22] J. Vygen, Algorithms for large-scale flat placement, in: Proceedings of the 34th Design Automation Conference, ACM, New York, 1997, pp. 746–751.
[23] J. Vygen, On dual minimum cost flow algorithms, Math. Methods Oper. Res. 56 (2002) 101–126.
[24] J. Vygen, Theory of VLSI Layout, Habilitationsschrift, University of Bonn, 2001.